NAVAL POSTGRADUATE SCHOOL
Monterey, California

THESIS

THE DEVELOPMENT OF A RELATIONAL DATABASE TO SUPPORT THE FLIGHT HOUR PROGRAM OF COMMANDER, NAVAL AIR FORCES PACIFIC

by

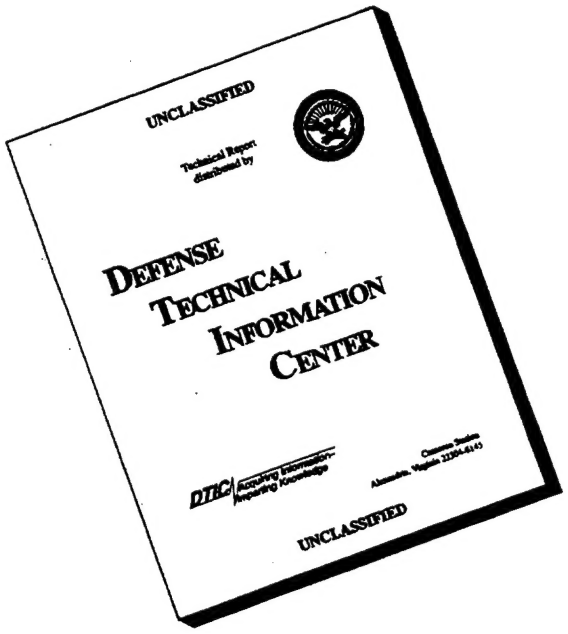
Mark J. Gonzalez
and
Mitch R. Hayes

June 1996

Thesis Advisor: C. Thomas Wu
Thesis Co-Advisor: John S. Falby

Approved for public release; distribution is unlimited.

DISCLAIMER NOTICE



THIS DOCUMENT IS BEST QUALITY AVAILABLE. THE COPY FURNISHED TO DTIC CONTAINED A SIGNIFICANT NUMBER OF PAGES WHICH DO NOT REPRODUCE LEGIBLY.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	June 1996	Master's Thesis

4. TITLE AND SUBTITLE
THE DEVELOPMENT OF A RELATIONAL DATABASE TO SUPPORT THE FLIGHT HOUR PROGRAM OF COMMANDER, NAVAL AIR FORCES PACIFIC

5. FUNDING NUMBERS

6. AUTHOR(S)
Mark J. Gonzalez and Mitch R. Hayes

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Naval Postgraduate School
Monterey, CA 93943-5000

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)

10. SPONSORING / MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES
The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

12a. DISTRIBUTION / AVAILABILITY STATEMENT
Approved for public release; distribution unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT ***(maximum 200 words)***

The Air Forces Pacific Financial Analysis Tool (AFAST) has been described by senior leadership within the Air Forces Pacific (AIRPAC) staff as inaccurate, marginally satisfactory and too small in scope to produce output in a relevant and timely manner. Moreover, improvements to the current database are time consuming and cost prohibitive; the absence of adequate design documentation prohibits major modification of the database without significant investment of the limited resources possessed by AIRPAC.

The primary goal of this thesis is the development of an improved conceptual design of the AFAST database based on the enhanced entity-relationship model concepts. The secondary goals of the thesis are the specification of the logical design of the improved database, and the implementation of AFAST II, a prototype application of the redesigned database.

The results of this thesis are: (1) an enhanced entity-relationship model that fully meets the design goals of Naval Air Forces Pacific, (2) the specification of the logical design for the implementation of the redesigned database, and (3) the development of a prototype application validating the conceptual and logical designs.

14. SUBJECT TERMS
Air Forces Pacific Financial Analysis Tool (AFAST)
Commander Naval Air Forces Pacific (AIRPAC)

15. NUMBER OF PAGES
148

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	UL

NSN 7540-01-280-5500

Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. 239-18
298-102

Approved for public release; distribution is unlimited

THE DEVELOPMENT OF A RELATIONAL DATABASE TO SUPPORT THE FLIGHT HOUR PROGRAM OF COMMANDER, NAVAL AIR FORCES PACIFIC

Mark J. Gonzalez
Commander, United States Navy
B.S., United States Naval Academy, 1980

Mitch R. Hayes
Lieutenant, United States Navy
B.S., Seattle University, 1988

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN COMPUTER SCIENCE

from the

NAVAL POSTGRADUATE SCHOOL
June 1996

Authors

Mark J. Gonzalez

Mitch R. Hayes

Approved by:

C. Thomas Wu, Thesis Advisor

John S. Falby, Co-Thesis Advisor

Ted Lewis, Chairman
Department of Computer Science

ABSTRACT

The Air Forces Pacific Financial Analysis Tool (AFAST) has been described by senior leadership within the Air Forces Pacific (AIRPAC) staff as inaccurate, marginally satisfactory and too small in scope to produce output in a relevant and timely manner. Moreover, improvements to the current database are time consuming and cost prohibitive; the absence of adequate design documentation prohibits major modification of the database without significant investment of the limited resources possessed by AIRPAC.

The primary goal of this thesis is the development of an improved conceptual design of the AFAST database based on the enhanced entity-relationship model concepts. The secondary goals of the thesis are the specification of the logical design of the improved database, and the implementation of AFAST II, a prototype application of the redesigned database.

The results of this thesis are: (1) an enhanced entity-relationship model that fully meets the design goals of Naval Air Forces Pacific, (2) the specification of the logical design for the implementation of the redesigned database, and (3) the development of a prototype application validating the conceptual and logical designs.

TABLE OF CONTENTS

LIST OF FIGURES

ACKNOWLEDGMENTS

We would like to thank Dr. C. Thomas Wu and John S. Falby for their time and guidance in the development of this thesis. Additionally, we would like to thank Phil Graessle and Chuck Anderson of AIRPAC for their technical assistance.

We also feel it is important to recognize the support of the staff at the Naval Postgraduate School. Thank you for your help.

I. INTRODUCTION

This thesis describes the design and development of a database which will be utilized to assist Commander, Naval Air Forces, Pacific (CNAP) with the management and analysis of aircraft cost data. The system discussed in this document is a major redesign of the database structure, graphical user interface and hardware architecture of the AIRPAC Financial Analysis Tool (AFAST), a database application created and implemented by the Air Forces Pacific (AIRPAC) staff. Additionally, this document presents our recommendations and results which summarize system design enhancements, prototype performance and the likely configuration requirements necessary to implement this design on the current AIRPAC network.

A. OVERVIEW

In 1994, faced with a shrinking budget and continued requirements to respond effectively to unique and diverse tasking, CNAP requested that a team be assembled to identify, analyze and assess the capabilities of the existing financial management systems being utilized in his enterprise. The objective was straightforward: "develop a methodology for operating AIRPAC in a more business-like manner" [Boyd94].

In the fall of 1994 selected members of the AIRPAC staff, augmented with contractor expertise, began the design, definition and implementation of AFAST. With limited resources and an optimistic operational target date of July 1995, the AIRPAC AFAST working group elected to create the database application using Foxpro, a commercial, off-the-shelf software application which would support the design of a

database, as well as the design of user interfaces, reports and graphic displays. In July 1995, Vice Admiral R.J. Spane (then, Commander Naval Air Forces Pacific) expressed concern that the application was not yet functional and requested that the Naval Postgraduate School (NPS) perform an independent evaluation of the AIRPAC work effort and utility of the AFAST application.

One of authors provided on-site assistance to AIRPAC for 2.5 months, during which time we:

- Voiced our support of, and reaffirmed the requirement for, AFAST.
- Expressed concern with regard to the absence of design documentation.
- Questioned the accuracy of imbedded algorithms.
- Confirmed the AIRPAC staff's speculations that the AFAST application design had performance limitations given the intended size and scope of the database.

In October 1995, the authors requested and received permission from AIRPAC to begin the redesign of AFAST. The scope of the initial research was limited due to the absence of funding and limited faculty support. In April 1996, after reviewing the NPS preliminary redesign proposal (as well as weighing system requirements and development costs), CNAP elected to fund the Naval Postgraduate School effort (to include faculty support), to provide an operational database, associated applications and system documentation.

B. MOTIVATION

The defense budget continues to decline and the armed forces are employed in increasingly unexpected roles, business processes are receiving increasingly more attention

from the Department of Defense, and more definitively, the United States Navy. While classic profit and loss accounting is not (always) applicable in a military enterprise, intelligent cash flow management is still vital. Careful and timely analysis of dollar expenditures coupled with rigorous review of money spent against operational tempo will enable AIRPAC to bring sensible business financial management practices to bear within the day-to-day decision making process. Cash flow analysis and management is critical to the success of CNAP in meeting its mission requirements. In order to manage AIRPAC more like a business and solve data distribution, quality and consistency anomalies, a data interface (i.e., database system), supporting the AIRPAC business process is required.

This reasoning fueled the motivation for the design of AFAST, a system prematurely declared operational in the summer of 1995. And this reasoning persisted as the impetus and motivation to fund NPS to develop and implement AFAST II.

C. THESIS OBJECTIVES

There are four objectives to this thesis. The first objective is to design the Entity-Relationship model which formally represents the network of relationships among the various AIRPAC commands which generate aircraft costs. The purpose of this design is to satisfy user requirements by graphically displaying the database structure, visually defining the conceptual design which will eventually be refined to yield the logical and physical structures which correspond to the implemented database.

The second objective is to develop an application, an interface between the user and the stored data. Recalling guidelines put forth by Nielsen [Nielsen90], the application should be:

- Easy to learn
- Efficient to use
- Easy to remember
- Generate few errors
- Be pleasant to use

The third objective is a culmination of the first two: implement and test a prototype using the designed database and application. It is the expectation that this prototype will be the framework for the database system to be delivered to AIRPAC, by NPS, in September 1997. The fourth and final objective of this thesis is to identify the commercial off-the-shelf software and hardware requirements necessary to construct a client/server configuration for AIRPAC's implementation of AFAST II.

D. ORGANIZATION OF THE THESIS

AIRPAC requirements and current system design deficiencies are presented in Chapter II to provide the reader with the necessary background required to grasp the importance of the design and implementation decisions which are presented in Chapters III and IV. Conclusions and recommendations are presented in Chapter V. Source code as well as glossaries, supplemental tables and charts, and system flow diagrams are contained in the appendices.

II. BACKGROUND

A. AFAST OVERVIEW

In 1994 "CNAP requested that a detailed and extensive Financial Management study be conducted to identify, analyze and assess capabilities of existing financial management systems currently utilized throughout the (AIRPAC) enterprise" [Boyd94 p. 2-1]. Key findings from the study include:

- Critical data required in the day-to-day decision making process was limited by availability, currency and accuracy.
- The absence of a Management Information System (MIS) organization severely hampered the collection and dissemination of relevant data.
- The inability of senior management to effectively monitor the AIRPAC spending process contributed significantly to the organization being managed in a reactive, crisis-oriented manner.

The select team of senior naval reservists conducting this study, all with significant management and business expertise, concluded their report by stating "In order to manage AIRPAC more like a business, and to implement the Financial Management Model while solving data distribution, quality and consistency problems, it is recommended that a data interface be constructed" [Boyd94 p.6-1]. As was presented in Chapter I of this document the focus of our effort is to improve upon and expand the capabilities of the "interface", AFAST, designed by this distinguished team. To do so requires an intimate understanding of the financial processes, methods and variables which influence the operation of AIRPAC. The sections that follow address fundamental

processes, all which must be captured in the eventual design of the new database. Specifically, we focus upon in the remainder of this chapter:

- AIRPAC financial management practices.
- Subordinate command reporting procedures.
- Database design requirements.
- Major deficiencies of the current (AFAST) database application.

B. FINANCIAL MANAGEMENT PRACTICES

AIRPAC's operational commitments and financial obligations are extensive. Responsible for aircraft, aircraft carriers, and air stations and facilities, CNAP must ensure readiness and operational requirements are met without overspending its annual budget. Analysis of the annual budget reveals that almost two-thirds of AIRPAC's financial obligations are destined for the support of aircraft maintenance and aircraft fuel costs. Collectively these costs fall within the bounds of the AIRPAC Flight Hour Program (FHP). It was the intended design of the (initial) AFAST application to model the processes associated with the FHP in order to provide a more timely and accurate estimate of funds expended. These processes - maintenance, supply, financial, Operational Report 20 (OP-20) allocation and flight hour allocation - collectively establish the bounds of the database design and define the structure which the database must assume. Their understanding is paramount to the successful development of the AFAST II database application.

1. Maintenance Process

The process is initiated with the identification of a failed aircraft part or system (Figure 1). Maintenance action is begun (at the local level) and documented using the Maintenance Action Form (MAF). At this early juncture an initial determination is made as to the level of effort required to correct the maintenance discrepancy.

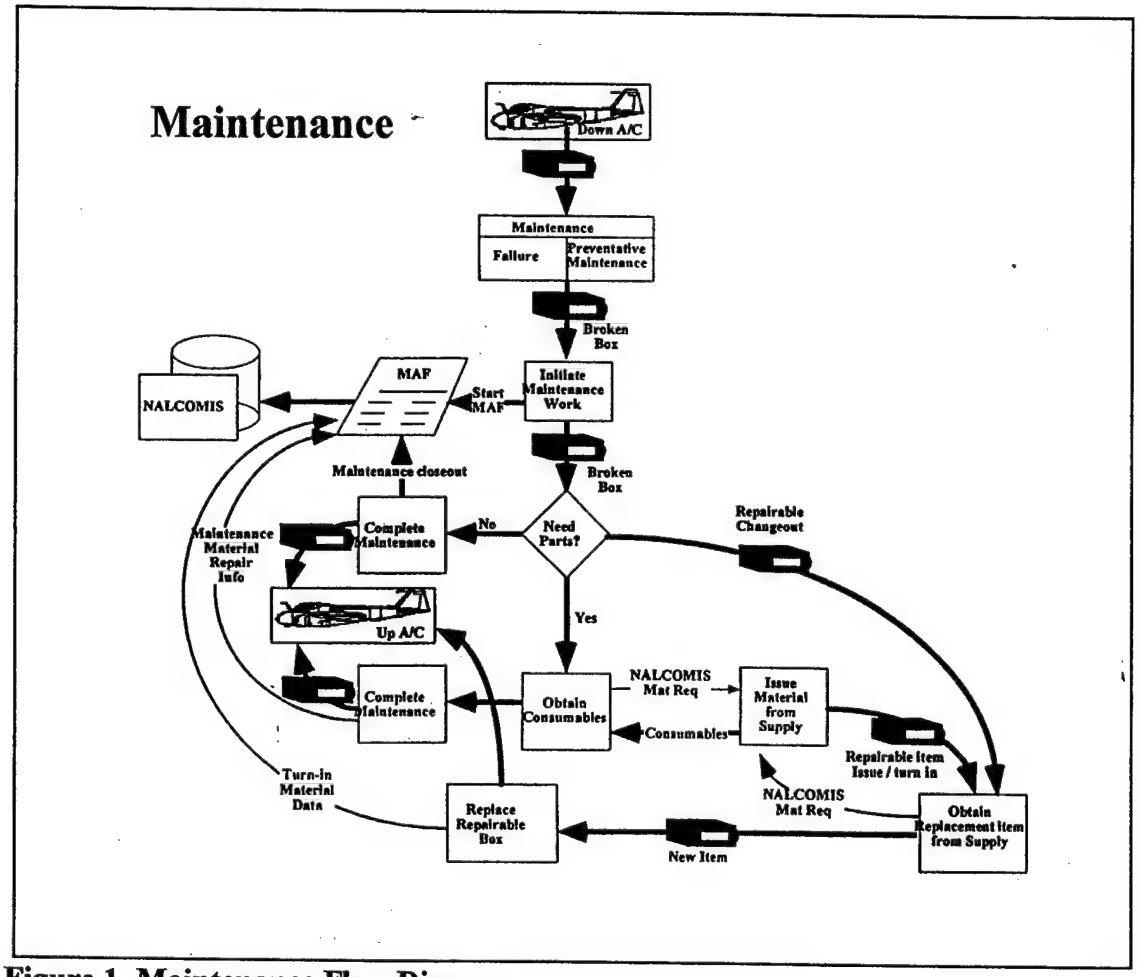


Figure 1. Maintenance Flow Diagram

Options are:

- The aircraft part (hereafter referred to as an assembly) can be repaired locally (an organizational maintenance effort) and reinstalled, in which case the discrepancy is corrected, documented on the MAF, and the MAF data entered into the Navy Aviation Logistics Command System (NALCOMIS) database (the significance and importance of NALCOMIS will be discussed in section C). The use of consumable materials (nuts, bolts, rags, cleaners, etc.,) required

to implement the repair are also recorded on the MAF, with these items being included in the data entered into the NALCOMIS database.

- Maintenance requiring replacement of an assembly defined as "repairable." The failed assembly is removed from the aircraft and forwarded to the local supply system as "non-ready for issue" (non-RFI). With receipt of the non-RFI assembly, supply issues an RFI replacement, and this assembly is then installed into the aircraft. The replaced "repairable" assembly is documented on the MAF, along with a listing of all maintenance performed to effect the removal and reinstallation of the assembly. All consumable materials required to implement the repair are recorded on the MAF, and as in the previous example, all MAF data is entered into the NALCOMIS database.

- Maintenance requiring replacement of an assembly defined as " nonrepairable." This will be discussed in the Supply Process.

At this stage the maintenance process is completed. It is important to note that while parts (may) have been exchanged and consumables have been expended, no money has actually been spent or obligated.

2. Supply Process

The non-RFI assembly (Figure 2) is delivered by the organizational maintenance activity's supply department to the local Aircraft Intermediate Maintenance Department (AIMD) or Intermediate Maintenance Activity (IMA) where a determination is made as to whether or not the assembly can be repaired locally at the AIMD. If repair is feasible at the AIMD level, the assembly is repaired and returned to the local supply system inventory (in the case of an AIMD aboard a carrier, if the assembly cannot be repaired locally, the assembly may be forwarded on to a shore-based AIMD where repair can be accomplished). In all instances corrective action taken and materials required to effect the assembly repair are documented on a supplemental MAF, and once again, all data are

entered into the NALCOMIS database. As was the case in the Maintenance process, there is no money spent or obligated, and the repaired assembly is received into the local supply system for future (re)issue.

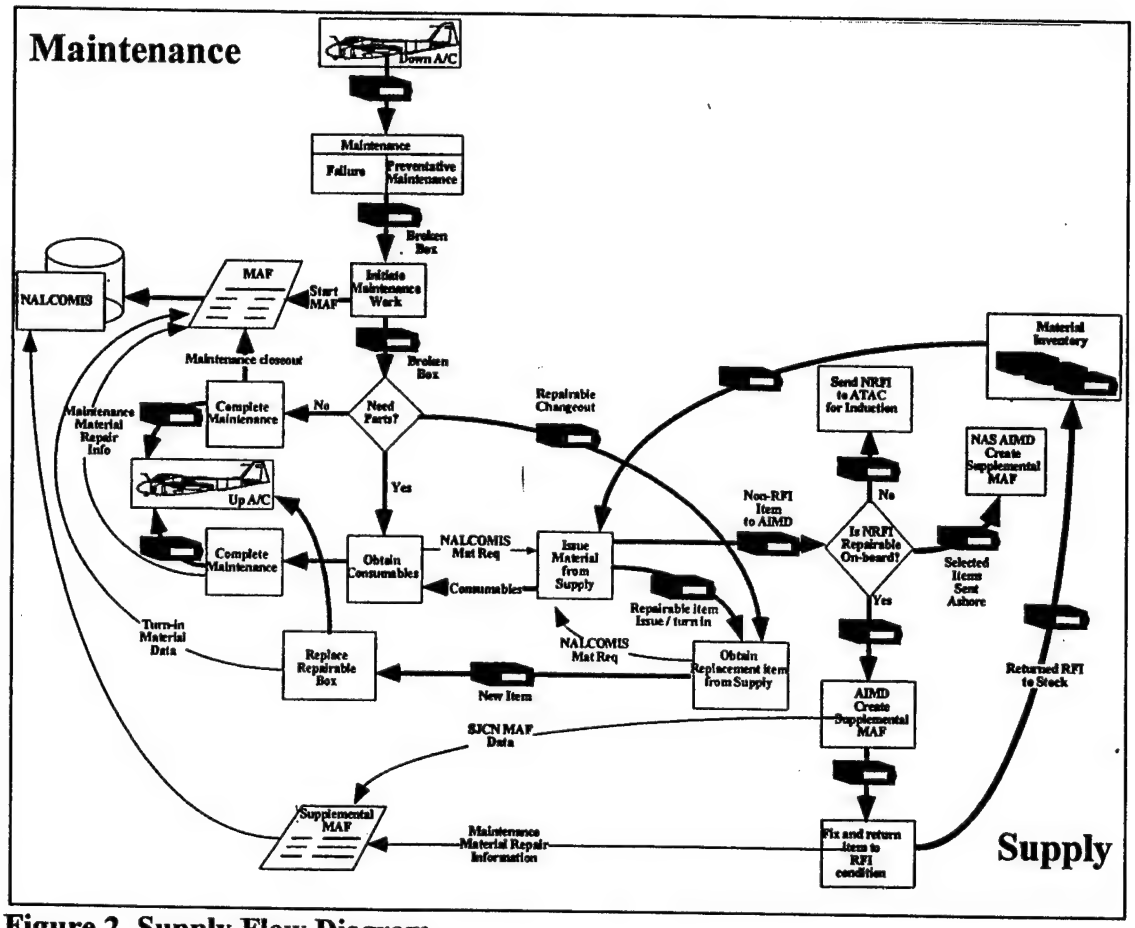


Figure 2. Supply Flow Diagram

Should the repair of the assembly be beyond the intermediate maintenance level capability, it is considered Beyond Capability of Maintenance (BCM) and the assembly is sent to an appropriate depot level repair activity. Local supply initiates a stock replenishment action (requisition) and the non-RFI item is processed through the local supply activity into the nearest Advance Traceability and Control (ATAC) hub, where credit for the carcass (the non-RFI assembly) is documented. In exchange for the non-RFI

assembly the local supply system is issued an RFI assembly, which is received into the local supply system and made available for future issue. Again, no financial transactions have occurred.

The description of the supply process is complete, however there do exist certain alternatives available to local supply departments which directly impact costs seen (or not seen) by AIRPAC. Local supply can forward for repair a non-RFI assembly but elect not to immediately reorder a replacement. By not reordering, actual maintenance costs as well as material consumption will remain invisible to AIRPAC. Only when a decision to reorder is made and a requisition (for a replacement assembly) generated are funds obligated (see Financial Process).

3. Financial Process

As depicted in Figure 3, once material is shipped from the Wholesale Supply System (WSS) to a requisitioning activity (i.e., a shore-based AIMD), charges for the assemblies are forwarded to Defense Accounting Activity which maintains account balances for AIRPAC and its subordinate commands. For repairable items, a "net charge" is billed, reflecting credit for turn in of the non-RFI carcass. If the carcass is not turned in and/or never received at the ATAC (see Supply Process), the requisitioning activity will eventually be charged for the missing carcass, and, as a result, is now charged the full price for the repairable assembly. Other debits, such as surcharges and price adjustments, are also billed to the requisitioning activity (through the Defense Accounting Activities), which is ultimately notified of the price adjustment via a document called the Summary Filled Order Expenditure Difference Listing (SFOEDL).

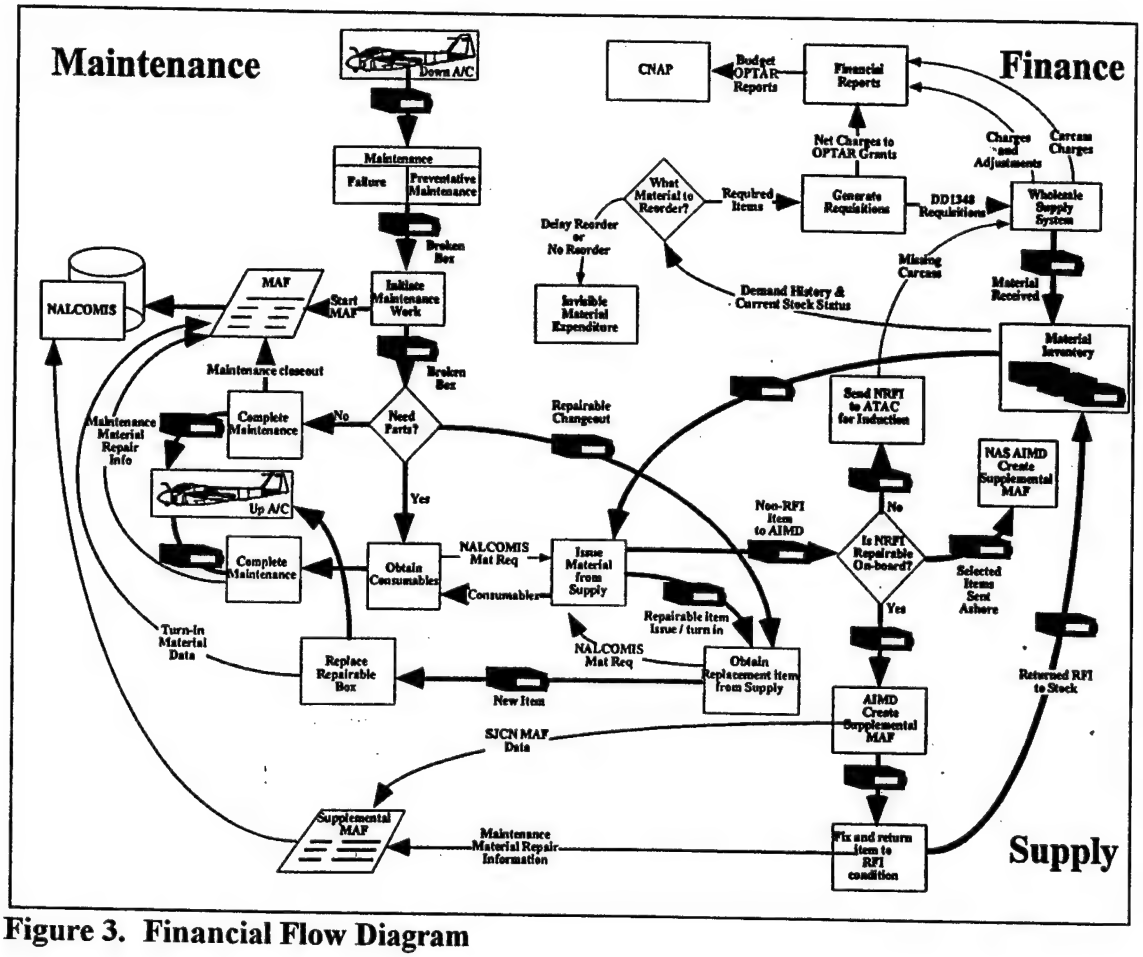


Figure 3. Financial Flow Diagram

It is only when material is procured from the WSS that financial obligations are incurred by subordinate activities and reported to the AIRPAC comptroller. This is done monthly via the Budget Optar Report (BOR), an official mandated report modified by AIRPAC to include flight hours, adjustments to obligated funds (i.e., surcharges and price adjustments) and any other additional information relevant to aircraft cost data. (Note: the BOR is the basis for official accounting records developed by AIRPAC. AFAST database does not utilize BOR data, but its output will be compared to BOR data to insure relative accuracy as well as corroborate data analysis conclusions drawn from the BOR database).

Implied but not yet directly addressed are the time lags which may be associated with the costs of performing maintenance. An aircraft may have a non-RFI assembly replaced by an AIMD which, as per the outlined procedure, documents maintenance performed in the NALCOMIS database. NALCOMIS generates a net-price, assuming:

- Non-RFI carcass will be returned
- Price data in NALCOMIS database is current
- No surcharges will be assessed on the assemblies replaced
- A replacement assembly will be purchased (if applicable)

As will be shown, it is cost data provided by NALCOMIS which are used by AFAST (and AFAST II). It is quite common to have prices associated with assemblies change, non-RFI assemblies not returned for months, or replacement assemblies not immediately (or ever) ordered. Unfortunately, it will be routine for AFAST cost data to differ from cost data provided by the BOR. (Note: this discrepancy is known to the sponsor. A determination on how best to resolve the cost anomalies is pending as this document goes to print. It is anticipated that the ultimate resolution will be incorporated into the AFAST II follow-on software build.)

4. Operational Report-20 (OP-20) Allocation

Figure 4 summarizes the processes described in the preceding sections and depicts the flow of funds from OPNAV (N88) through Commander-in-Chief Pacific Fleet (CINCPACFLT) and into the coffers of CNAP. The flight hour budgeting process is the basis for granting funds to CNAP activities in support of aircraft operations and

maintenance [CNAP86]. The document governing the transfer of funds is the OP-20, a report which is the basis for the Flight Hour Program Budget. The budget cycle drives the report promulgation dates and normally three major revisions can be expected during a fiscal year. The OP-20 report promulgates:

- Budgeted flight hours.
- A delineation of Aircraft Flight Operation (AFO) costs, Aviation Depot Level Repair (AVDLR) costs, and Aviation Fleet Maintenance costs (AFM) in terms of a projected average fleet-wide Cost Per Flight Hour (CPH).
- Annual costs for each type/model/series (TMS) aircraft assigned to AIRPAC.

The AIRPAC staff utilizes the OP-20 to assist them in making the allocation of flight hours to each squadron/wing/aircraft-owning activity, taking into account deployment schedule, training requirements, etc. An example of an OP-20 report is provided in Appendix B. When allocation levels have been determined, the AIRPAC Flight Hour Program Office will "distribute", usually on a quarterly basis, the flight hours via an OPTAR Grant message to the various squadrons/wings/aircraft-owning activities. This distribution process creates a "budget" for each squadron, however, actual maintenance funds (as depicted in Figure 4) are allocated to the supporting CV's and Naval Air Stations which maintain the books for these squadrons. An example:

AIRPAC allocates VFA-122 1000 flight hours for quarter one of fiscal year 1995. During the allocation process the AIRPAC staff estimates that, based upon the upcoming deployment schedule of VFA-122, 600 hours will be flown at NAS Lemoore and 400 hours will be flown while operating aboard the USS Kitty Hawk. Therefore AIRPAC

would allocate VFA-122's AVDLR and AFM funds to NAS Lemoore and USS Kitty Hawk based on the expected levels of flight operations at each installation.

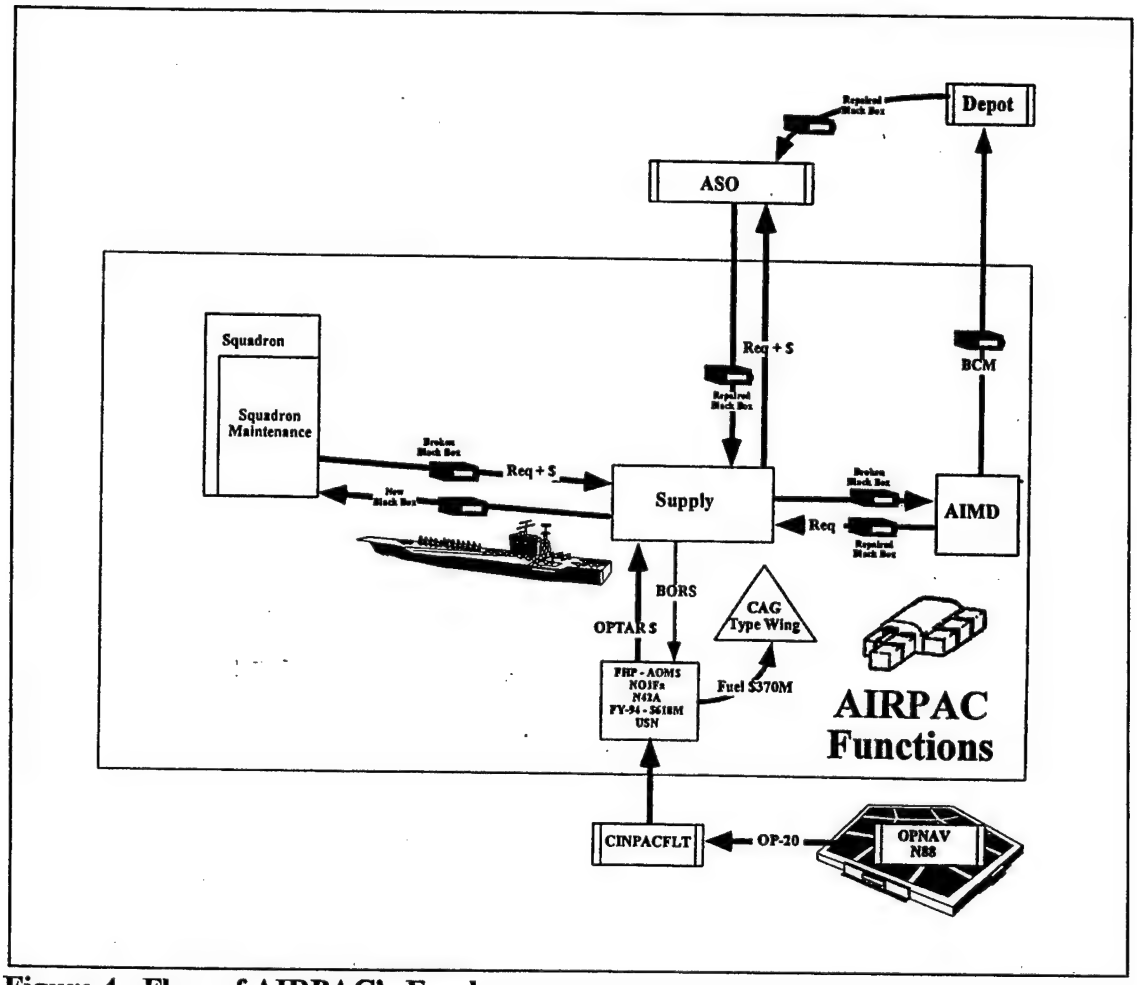


Figure 4. Flow of AIRPAC's Funds

Note: OPTAR grants can be and are frequently changed and reprogrammed as operations change and the level of support provided to a squadron by an AIMD activity changes from the initial budget assumptions.

C. SUBORDINATE COMMAND REPORTING PROCEDURES

Aircraft flying hour and maintenance costs are forwarded by subordinate activities to AIRPAC via two reports: the Aviation Storekeeper Information Tracking System

(ASKIT) AFAST report, and the NALCOMIS AIMD Cost Accounting (NACA) extract program. These reports, designed to support other databases, are processed and artfully manipulated to output the raw data which comprises the bulk of the AFAST database.

1. ASKIT AFAST Report

The ASKIT system [CNAP93] is designed to assist squadrons with material requisitioning, requisition tracking, and financial data processing and management. Provided within the ASKIT system is the semi-automated AFAST report generation function which outputs bi-monthly the following data:

- Reporting activity
- Report Type (initial report, year-to-date summary, weekly report, update/change report)
- Reporting period
- For each TMS assigned to the squadron:
  - Flight hours flown
  - Fuel costs
  - Location (Home/Deployed) where flight hours were executed

All aircraft-owning activities submit the report to AIRPAC where the data is processed and compiled by AFAST for inclusion into the database. The resulting actual flight hour cost data can then be compared with AIRPAC allocation data and OP-20 cost figures to document trends and identify expenditure anomalies. A sample ASKIT AFAST report is provided in Appendix B.

2. NALCOMIS AIMD Cost Accounting (NACA) extract program report

The NACA report was developed to track AOM funds using data extracted from the NALCOMIS database. The report is compiled by AIMDs, forwarded to AIRPAC monthly, and provides a means of monitoring the cost of assembly requisitions as they are generated by squadrons and the reporting repair facilities. Originally designed as a tool for the AIMDs, NACA data is incorporated into the AFAST database where it can be extracted and used to monitor AVDLR and AFM costs associated with the various type aircraft assigned to AIRPAC. As was the case for the fuel costs and flight hour expenditures, the maintenance cost data provided by NACA can be contrasted with AIRPAC allocation data and OP-20 cost figures to document trends and identify expenditure anomalies. However, NACA cost data is only approximate and assumes that (non-repairable) assemblies will be returned to the appropriate depot level repair facilities, and replacement assemblies will be ordered by the applicable supply department. It does not account for:

- Changes in assembly prices.
- Assemblies that are not ordered.
- Assembly carcasses that are not forwarded to Depot Level Repair Activities.
- Assemblies that are ordered, and later canceled.

(Note: the deficiencies associated with NACA are significant. It is anticipated that upgrades to AFAST II will be required to correct these inaccuracies by providing interfaces with other navy maintenance management systems.)

A reporting activity's monthly NACA report is received in the form of two ASCII text files (Appendix B), both of which are used to archive requisition data and record completed maintenance costs. The NACA report is unique in that a maintenance action may be initiated (i.e., recorded) in one monthly report, yet work may continue on that specific action for a significant time period (i.e., 2-6 months). Interim costs and maintenance action resolution will then be reported via ensuing NACA reports; it is critical that the database be able to accurately track individual maintenance actions through completion as well as be able to identify those actions that remain pending.

D. DATABASE DESIGN REQUIREMENTS

1. Overview

The AFAST II application design has been highly influenced and considerably constrained by the original AFAST system developed by AIRPAC. Having expended significant amounts of time, research and funding towards defining and implementing the system, it was expected the initial solutions garnered by the original effort would be refined and incorporated (by NPS) into AFAST II. In retrospect, the initial efforts focused primarily on design goals and high level requirements, paying little attention to data specifics, application tools, and feasibility of design. The consequence proved fatal as small-scale commercial database software tools were chosen to implement the database, and, when coupled with the prevailing hardware architecture, an application with marginal performance and questionable accuracy resulted. In December 1995 [Gonzalez95] it became apparent that AFAST needed to be re-engineered; the authors accepted the tasking and began the bottom-up review and redesign of the system.

2. Definition and Scope of AFAST II

The purpose of AFAST II is to receive, process and display aircraft maintenance, flight hour/fuel and supply system obligation data. The system relies on three sources of data to populate the database. The first two:

- ASKIT report
- NACA report

are dynamic data sources, submitted at regular intervals by CNAP subordinate commands, providing a mechanism to capture costs as they are generated by squadrons and AIMDs. The third source of data is, in the general sense, characterized as static and is reference data used by the application to process and store the ASKIT and NACA reports. It includes:

- Squadron data
- Organizational data
- Naval Air Station and Carrier data
- OP-20 allocation data
- CNAP allocation data

The preponderance of the static data is provided by AIRPAC Flight Hour Program Office; the common thread amongst this data is its constancy. It remains relatively stable during the course of the fiscal year, changing only if and when subordinate activities (squadrons, organizations, CVs...) relocate or are decommissioned, or when allocation data is modified.

As previously mentioned the original database was built using a small-scale software tool which not only impacted the application's utility but limited its availability. AFAST II overcomes this deficiency by taking advantage of the current AIRPAC hardware architecture and configuring the application in a client/server environment. The design provides a cost-effective approach to adding value to the organization (with only minimal software investment) making a single database of financial information available to all within the enterprise. The hardware configuration will be such that client and server processes are executed on separate machines, connected via a network, with one database server supporting multiple clients. Per tasking from the AIRPAC staff, the database will be located in a Sun SPARC server, networked to its clients using Novell netware and an appropriate middleware. Additionally, the physical and security constraints associated with the location of the server dictate that the database administrator's primary access to the database (to include metadata and proprietary procedures) be via a client station.

Implementing AFAST II on a client/server architecture greatly increases the number of primary users. Originally limited to senior AIRPAC staff the new design makes the system available to:

- AIRPAC middle managers
- Subordinate squadrons, maintenance facilities and staffs

The increase in the numbers and types of users has necessitated the evaluation and subsequent redesign of the methods used to select and display data. AFAST II departs from the original system by designing the system with the customer, the Naval Aviator, as the focal point. Stricken or severely minimized are irrelevant and unfamiliar references to

financial management practices. In their place are common and more apparent definitions of the data elements, and more user-appropriate accesses to the data. The end result is that a greater number of users will be allowed to easily navigate through the application and readily access wanted data.

3. Database Design Feasibility

Three elements are addressed in the AFAST II feasibility assessment. These are: software availability, hardware availability and time constraints.

a. Software Availability

Numerous commercial software application tools were available which provided both the capability to design views (user applications) and databases. The AIRPAC staff, in efforts to promote commonality and reduce personnel training costs, strongly encouraged NPS to develop AFAST II using ORACLE7® as the database management system (DBMS). It was readily determined that this software would easily meet the design requirements, however, a delay in funding required the authors to explore other, interim, alternatives. The software strategy was to procure, as economically as possible, a software tool which would:

- Produce high quality user applications
- Closely image the programming features provided by ORACLE, with the intent to transition to ORACLE7® when funding became available.

Based on preliminary research and the analysis of small-scale database design projects [Wu 96], it was determined that the Delphi Desktop rapid application development tool

best met the requirements and still fell well within the research budget constraints. (Subsequent to the purchase of the Delphi tool and the commencement of database design, AIRPAC funded the NPS effort; necessary modifications of the software design to meet CNAP objectives are delineated in Chapter 5, Conclusions and Recommendations.)

b. Hardware Availability

The hardware required to drive the Delphi software ranges from desktop personal computers (IBM PCs) to workstations (SUN/Sparc). Within the scope of this thesis, hardware will be limited to an IBM PC (acting as both client and server); given the portability of the Delphi application all development work will be easily transferred to the AIRPAC architecture which employs a SUN/Sparc server to support multiple (IBM PC) clients. Any necessary modifications of the AIRPAC hardware configuration to meet CNAP objectives are delineated in Chapter 5, Conclusions and Recommendations.

c. Time Constraints

At the time funding was provided by AIRPAC work had already commenced on the Delphi application. It was determined that this effort would continue until June 1996, at which time the resulting prototype would begin modification, as necessary, to be implemented on the AIRPAC architecture. Delivery of the AFAST II, using ORACLE7® as a DBMS, and configured to utilize a SUN/Sparc server would be required no later than September 1997. Documentation of this work effort will be provided under separate cover, as the scope of this document is limited to the development of the Delphi AFAST II prototype.

E. DEFICIENCIES OF CURRENT (AFAST) DATABASE APPLICATION

As stated in the introduction, NPS agreed to not only provide the same functionality as the current AFAST application, but also correct documented design deficiencies associated with the system. Based on initial analysis and tests conducted by NPS (and verified by AIRPAC), the following seven deficiencies can be classified as major and must be corrected in AFAST II.

1. Absence of Documentation

There exists minimal documentation which defines:

- Design decisions
- Algorithms and procedural functions
- Database model
- Domain restrictions

The scarcity of design documents has slowed AIRPAC's ability to verify data accuracy and rectify errors in the original application. The nonavailability of documentation has also impacted AFAST II design efforts in that there does not exist a requirements document which directs the manner and method that complex calculations and business rules will be implemented. Significant energy has been devoted to revisiting requirements with AIRPAC, as "Incomplete, misinterpreted or unrealistic requirements are many times the root cause of...project failures."[Pressman92]

2. Squadron Deployment Restrictions

The original design assumed, incorrectly, that all squadrons assigned to CNAP deployed to carriers. Additionally, it was assumed a squadron could deploy to only one carrier during the course of a fiscal year. The impact of these design flaws is significant. Costs generated by squadrons which deployed to sites other than carriers, or which deployed more than once during a fiscal year, could not be accounted for accurately in the database. The result was inaccurate cost estimates at all levels throughout AIRPAC subordinate activities which supported deploying squadrons.

3. Squadron Detachments

The AFAST application is unable to accurately track costs associated with squadrons which deploy detachments. Flight hours flown and costs incurred by the detachments are entered into the database using inconsistent and irreconcilable processes, none of which have documentation to support their use. The effect of this error is, once again, global, affecting not only the specific squadrons but the air stations and carriers which support them.

4. Squadrons with more than one type aircraft

The database design in the current system prohibits data being graphically displayed for squadrons which have more than one type/model/series (TMS) aircraft. The source of the deficiency is embedded in the database model; efforts to correct this deficiency require a total redesign of the database. The capability to graphically display

data was one of the highest priorities of the original system, and remains at the top of the list for AFAST II.

5. Database redundancy and design inefficiency

To overcome some of the early design problems encountered with the original database, the decision was made to add autonomous entities and relationships (i.e., tables) to the model rather than redesign, normalize and otherwise make more efficient the current database model. The end product is now a database which requires redundant data input, utilizes significantly more storage space than required, and processes data at a level significantly less than that which is acceptable [Lauff96].

6. Processed data accuracy

Minimal effort has been put forth on the part of AIRPAC to verify that stored procedures and client-side data calculations are accurate. Preliminary testing by NPS during the development of the AFAST II prototype suggests that significant errors exist in the computation and subsequent storage of processed data. As previously mentioned, little documentation exists to verify or validate the performance and results of the current system.

7. Graphical User Interface design and implementation

The current AFAST application is recognized by its use of single windows, clumsy and confusing procedures associated with accessing data, and the absence of standardization between individual application windows and between the application itself

and human-computer interface standards recognized and implemented by the computer industry. The task here was to redesign the user views from the ground up and employ accepted evaluation and usability guidelines in the design of the AFAST II user views.

III. DATABASE DESIGN

The development of AFAST II adhered to the guidelines put forth by Kroenke and Dolan [Kroenke88] who recommend utilizing a two-part design process when constructing a database system. The two parts are:

- Development of the database design
- Development of the application design

This chapter focuses on database design, specifying the specific constraints, requirements and design decisions which culminated in the development of the entity-relationship model, the blueprint of the database structure. It is necessary, however, to precede this unveiling with a brief overview of basic database terminology and fundamentals as these will be referenced throughout the remainder of this document.

A. TERMINOLOGY

1. Database

A database is a collection of related data possessing the following implicit properties [Elmasri94]:

- A database represents some aspect of the real world, oftentimes referred to as the miniworld. Changes to the miniworld are reflected in the database.
- A database is a logically coherent collection of data with some inherent meaning. A random assortment of data cannot correctly be referred to as a database.

- A database is designed, built and populated with data for a specific purpose. It has an intended group of users and some preconceived applications in which these users are interested.

A database must have direct access to a storage medium to enable processing, and, implemented with a suitable storage medium, has the capability to manage large integrated data structures in a shared multi-user environment. Physically the database consists of tables. However, the data is structured with only limited consideration given to table boundaries. Important objectives in the ultimate design of the database are the achievement of an extremely flexible level of interaction between small data items (attributes) resident in the tables, and a minimization of redundancy. When designed well, the database allows concurrent processing, maintains consistency of data, and supports testing and maintenance[Maciaszek89].

2. Database Management System (DBMS)

A DBMS is a collection of software programs that enables users to define, construct and manipulate databases for various applications. Defining a database involves specifying data types and domains for the data to be stored. Constructing the database is the method of storing the data on some specified storage medium that is managed by the DBMS. Manipulating a database encompasses the following:

- Querying the database
- Updating the database
- Generating reports from selected data

3. Entity-Relationship Diagram

"The cornerstone notation for data modeling is the entity relationship diagram" [Pressman92]. It is central to the conceptual design, providing a graphical representation of:

- Entities - "things" that can be distinctly identified. Any distinguishable object, real or abstract (i.e., squadrons, budgets).
- Attributes - particular properties associated with entities (i.e., squadron name, fiscal year).
- Relationships - an association among entities (i.e., squadrons *deploy on* carriers, NAS's *submit* maintenance transactions reports).

First introduced in 1976, the goal of entity-relationship modeling is to deliver an abstract, nonredundant and unified representation of the data of the application. With this model the network of relationships among the entities and their attributes can be clearly and explicitly expressed [Fidel89].

An uncomplicated symbol format is used to construct the entity-relationship model (Figure 5). A rectangle represents an entity. Lines with a "diamond shaped box" serve to connect entities, while the diamond symbols represent relationships between entities. Many different types of relationships exist, only two of which are of concern within the scope of this database application:

- one-to-many - represented by 1:M. Ex: A squadron can have *many* reports, however a report can have only have *one* squadron.
- many-to-many - represented by M:N. Ex: A squadron can deploy to *many* carriers, and a carrier can have *many* squadrons.

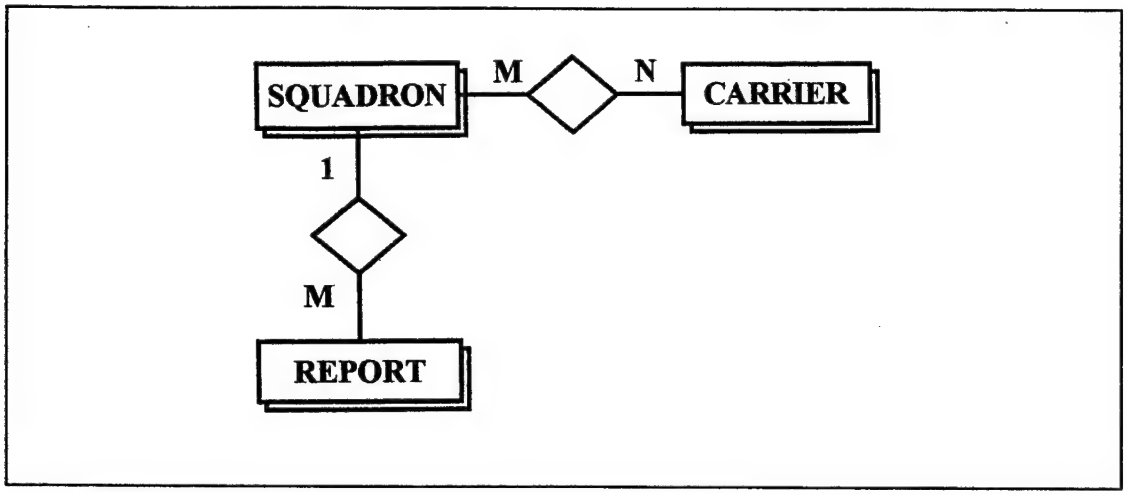


Figure 5. A simple Entity-Relationship Diagram

Using standard database design algorithms the entity-relationship model can be transformed into the relational database schema, in a general sense a collection of tables with each table uniquely representing each entity and relationship (Figure 6). Rows in the table represent a collection of related data values. Each row in the table is called a tuple, and is a representation of the facts about a particular record contained in that table. The table name and column names are used to aid in interpreting the meaning of values in each row of the table. The column names specify how to interpret the data values in each row based on the column each value is in. All values in a column are of the same data type (i.e., integer, character, date).

Name	UIC	Service	Mission	Type
VFA -137	R12345	USN	Tactical	VFA
VA -52	R54321	USN	Tactical	VA

Figure 6. A simple Relational Table

B. AFAST II ENTITY-RELATIONSHIP MODEL DESIGN

A database design process must be conducted within the framework of the enterprise's strategic plan. One of the aims of strategic planning is to determine the strategy for the development of a new information system. The high level strategy associated with the development of the AFAST II database was quite simple:

- Review the database structure of the original database (AFAST)
- Identify deficiencies with the current database
- Revise the database structure to overcome deficiencies

1. AFAST II Data Requirements

An in-depth review and analysis of the original database was not possible as system documentation was not available or was incomplete. A thorough survey of requirements documents and interviews with domain experts was conducted to specifically determine the requirements and clarify requirements that were misinterpreted and implemented in the original AFAST application. From this initial data collection effort the following specifications for the AFAST II database were validated and/or incorporated:

- Allocation data for squadrons, CV's and NAS's from the OP-20 report and CNAP is to be displayed.
- Data will be displayed in three different time periods: monthly, quarterly, yearly.
- Maintenance and flight hour cost data will be displayed for carrier air wings, squadrons and aircraft TMS's.
- Summary maintenance cost data will be displayed for NAS's and CV's. Data will be able to be subdivided to show costs generated by specific squadrons and tenant commands (organizations) attached to the specific CV or NAS.

- A summation view of all maintenance and flight hour cost data will be provided; display will correlate all costs incurred by CNAP.

- Specific maintenance costs will be displayed: These include categories AVDLR and AFM, and subcategories Engines, Airframes, Avionics, Miscellaneous, and Other (to be defined).

- Data will be displayed both graphically and in tables. Methods of display for specific views to be determined by designers.

2. AFAST II Entities

Once the requirements were collected the actual construction of the Entity-Relationship model began. Identified as entity types were the following:

- Squadrons
- Squadron Detachments
- Aircraft Type/Model/Series
- Commander, Naval Air Forces Pacific
- Naval Air Stations
- Carriers
- Carrier Air Wings
- Fiscal Year Budget (OP-20 Report)
- Fuel/Flight Hour Reports (ASKIT)
- Maintenance Transaction Reports (NACA)
- Organizations

It was noted early in the design phase that carrier's and naval air stations, within the scope of the database specifications, were identical, that is they shared all the same attributes. Given this condition it was necessary, in order to eliminate redundancy and optimize the design of the model, to group together carriers and NAS's into one entity. This action became the first of many significant departures from the design of the original model. As will be seen in the discussion of relationships, this restructuring provided the AFAST II database with greater flexibility and significantly more capability.

In December 1995 the AIRPAC AFAST working group determined that, at the present time, it was not a requirement to be able to display squadron detachment data. Rather, the costs generated by the detachments would be rolled into the costs of their parent squadrons. The exceptions to this new specification were detachments from VQ-5 and VRC-30. These detachments would be modeled as squadrons with only their deployed costs being presented on their individual data page. Costs which these detachments incurred while located at their home station (i.e., NAS North Island) would be presented as part of the parent squadron data.

One last entity was deleted from the original list of ten, that being Commander, Naval Air Forces Pacific (CNAP). After analyzing the attributes associated with each entity type and the relationships which existed between the entities, it was determined that CNAP views and associated data could be generated by summing data resident in the NAS/CV and squadron entities. The specifications associated with each entity are presented below:

- Carrier Air Wings - specifies air wings in the AIRPAC chain of command. CNAP requirement exists to display aircraft cost data by airwing grouping, this entity is the mechanism which enables that action.

- Fiscal Year Budget - defines amount of funding to be provided to CNAP to support flight hour and maintenance costs for each type aircraft. Funding levels vary depending on type service (USN or USMC) and mission (Tacair, Support, Fleet Readiness). Data is provided by OP-20 report.

- Fuel/Flight Hour Reports - reported by squadrons and other CNAP commands who are custodians of aircraft. Summarizes flight hours flown and fuel costs during a specific reporting period for each type aircraft owned by the squadron. Data is currently provided via the ASKIT report.

- Maintenance Transaction Reports - submitted by CV and NAS AIMD facilities who participate in the NACA program. A monthly report which details maintenance performed on aircraft by squadrons, CV's and their tenant commands, and NAS's and their tenant commands.

- NAS/CV - specifies current NAS's and CV's in the AIRPAC chain-of-command. CNAP requirement exists to display aircraft cost data by specific NAS or CV. This entity is the mechanism which enables that action, as well as enabling the tracking of costs generated by NAS and CV tenant commands which do not own aircraft.

- Organizations - tenant commands located onboard NAS's and CV's which generate costs associated with the maintenance of aircraft. These costs are reported in NACA and are eventually equated to specific squadrons and TMS's in order to determine aircraft operational costs.

- Squadrons - specifies current Squadrons in the AIRPAC chain of command. CNAP requirement exists to display aircraft cost data by specific squadron. This entity is the mechanism which enables that action, as well as enabling the presentation of costs associated with each type aircraft flown by a squadron.

- Type/Model/Series - delineates current types of aircraft CNAP is funded to operate and maintain as per the OP-20. CNAP requirement exists to display individuals as well as summary data for TMS costs.

3. AFAST II Entity Attributes

Concurrent with the entity resolution activity was the defining of attributes which were associated with each entity type. In most cases the attributes were determined by reverse engineering the original AFAST application (entities and their attributes are

presented in Appendix A). However there do exist certain unique entity attributes whose purpose requires description.

- ID_ TMS (in the TMS entity) - The purpose of this attribute is twofold. The first is to serve as aggregate for the following attributes within the TMS entity: TMS, Service, Mission and Fiscal Year. The second is to act as the key, the single attribute whose value is used to identify uniquely a record from all other records belonging to the particular entity, and whose function is to permit the accessing of other tuples stored in other tables. Use of the aggregate key saves storage space, reduces redundant data and improves performance by permitting other tables to use the aggregate attribute as a foreign key [see Elmasri94] rather than having to use four previously mentioned attributes.

- ID_ASKIT/ ID_NACA (Fuel/Flight Hour Entity/Maintenance Transaction Entity) - The purpose of this attribute is to function as a key for the records in each of the entities. It was determined early in the analysis of data contained in both the ASKIT and NACA reports that the reports did not provide a simple and consistent method by which each report could be identified, stored and processed uniquely. The ID_ASKIT and ID_NACA attributes accomplish this by assigning a unique integer value to each report. In essence the basic nature of these attributes is, simply, to facilitate record keeping.

4. AFAST II Relationships

The relationships used to produce the AFAST II model and link the entities are simple, straightforward and easy to resolve. A total of twelve different relationships are employed to link the eight entities; each relationship with its own unique characterizations or criteria which are used to strengthen and enhance the integrity of the E-R model. The dozen relationships created to support the E-R model (Figure 7) can be classified according to three criteria: degree, connectivity, and membership. The criterion of degree refers to the total number of entities linked by, or participating in, the relation. In all cases but one, the degree of AFAST II is two; the one exception is degree three, that is, there are three entities which are linked by one relationship.

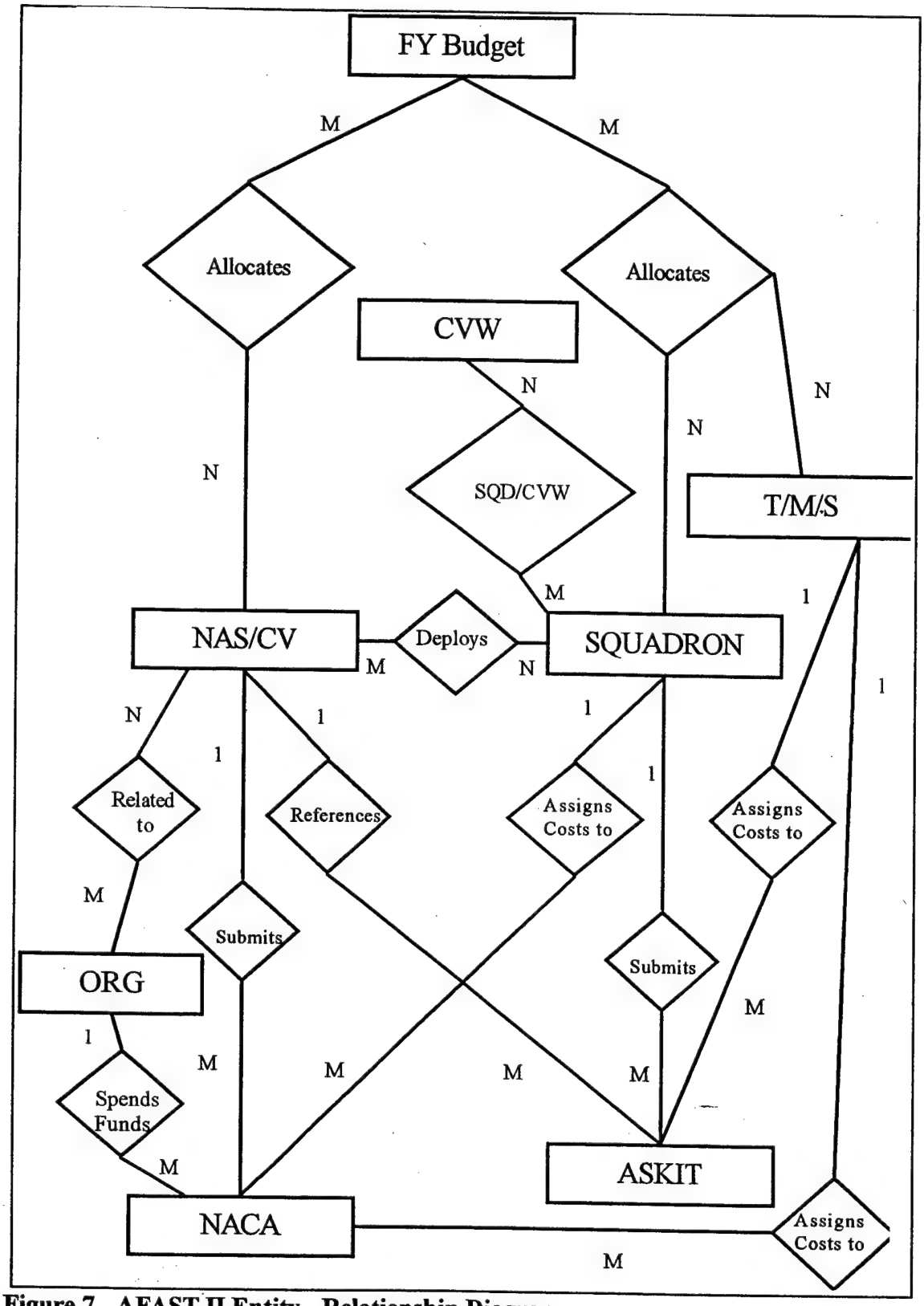


Figure 7. AFAST II Entity - Relationship Diagram

According to the criterion of connectivity the options available are singular and multiple relationships. Singular relationships are not employed in the AFAST II model; as previously discussed the relationships between the entities will either be 1:M or M:N.

With regard to the membership criterion, there are two types of participation constraints which are illustrated by example. In the AIRPAC paradigm, since all carrier air wings must have squadrons, then it can be said an air wing can only exist if it participates in the squadron/cvw relationship (Figure 7). The participation of carrier air wing in the squadron/cvw relationship is called total, meaning that every entity in "the total set" of carrier air wing entities must be related to at least one squadron entity via the squadron/cvw relationship. Again referring to Figure 7, it is not the case that every maintenance transaction report references an organization, so the participation of organization in the spends_funds relationship type is partial, meaning that some or "part of the set of" organization entities are related to a maintenance transaction report via spends_funds, but not necessarily all. The specifications associated with each relationship are presented below:

- Allocates - This relationship is used twice. In the first instance it connects the entities Fiscal Year Budget and NAS/CV, in the second instance it links Fiscal Year Budget with SQUADRON and TMS. In the first case the relationship is of degree two and requires total participation of both NAS/CV and Fiscal Year Budget. Being a multiple M:N relationship requires that when the standard database design algorithms are used to transform the entity-relationship model into the relational database schema (Figure 8), this relation will become a table with its own unique attributes. This relation hold allocated maintenance funding data for the FY BUDGET-NAS/CV relation. The second instance of Allocates is the only use of a relationship which has degree greater than two, in this case the degree is three. The need exists to establish a three-way relationship that binds together FY BUDGET, SQUADRON and TMS. This enables us to extract from the model for each squadron the TMS's allocated to it and the funding available to support each TMS. As was the case with the first instance participation is total, each SQUADRON and TMS having to be allocated funds

in order to exist as an entity instance. Being a M:N relationship, this relation will become a relational table with its own unique attributes. In this case the relation stores allocated maintenance funding data for FY BUDGET-SQUADRON-TMS.

- Assigns_costs_to - This relationship is used on three occasions: in the first instance it connects the MAINTENANCE TRANSACTION REPORT and SQUADRON entities; in the second instance it connects the MAINTENANCE TRANSACTION REPORT and TMS entities; and in the final instance it links FUEL/FLIGHT HOUR REPORTS with TMS. In all three cases the relationship is of degree two and requires total participation of the MAINTENANCE TRANSACTION REPORT and links FUEL/FLIGHT HOUR REPORTS. Being a multiple 1:M relationship requires that when the standard database design algorithms are used to transform the entity-relationship model into the relational database schema (Figure 8), this relationship's attributes will be combined with the attributes of the MAINTENANCE TRANSACTION REPORT and FUEL/FLIGHT HOUR REPORT entities. However, in this case there are no attributes associated with the relationships, so there is no requirement to merge attributes into these tables.

- Deploys - This relationship is used once. It links NAS/CV and SQUADRON entities. The relationship is of degree two and requires total participation of the SQUADRON entity. Being a M:N relationship, this relation will become a relational table with its own unique attributes. Deploys identifies specific SQUADRONS which are which are assigned to specific NAS's and CV's.

- References - This relationship is used once, it links NAS/CV and FUEL/FLIGHT HOUR REPORT entities. The relationship is of degree two and requires total participation of the FUEL/FLIGHT HOUR REPORT. Being a multiple 1:M relationship requires that when the standard database design algorithms are used to transform the entity-relationship model into the relational database schema (Figure 8), this relationship's attributes will be combined with the attributes of FUEL/FLIGHT HOUR REPORT entity. However, in this case there are no attributes associated with the relationship, so there is no requirement to merge attributes into the FUEL/FLIGHT HOUR REPORT table.

- Related_to - This relationship is used once. It links ORGANIZATION and NAS/CV entities. The relationship is of degree two and does not require total participation of either of the two entities. Being a M:N relationship, this relation will become a relational table with its own unique attributes. The relation identifies specific organizations which are permitted to spend the allocated funds of specific NAS's and CV's.

- Spends_funds - This relationship is used once. It links ORGANIZATION and MAINTENANCE TRANSACTION REPORT entities. The relationship is of degree two and does not require total participation of either of the two entities. Being a multiple 1:M relationship requires that when the standard database design algorithms are used to transform the entity-relationship model into the relational database schema (Figure 8), this relationship's attributes will be combined with the attributes MAINTENANCE TRANSACTION REPORT entity. However, in this case there are no attributes associated with the relationship, so there is no requirement to merge attributes into MAINTENANCE TRANSACTION REPORT table.

- Submits - This relationship is used twice: in the first instance it links the MAINTENANCE TRANSACTION REPORT and NAS/CV entities; and in the second instance it links FUEL/FLIGHT HOUR REPORTS with SQUADRONS. In both cases the relationship is of degree two and requires total participation of the MAINTENANCE TRANSACTION REPORT and links FUEL/FLIGHT HOUR REPORTS. Being a multiple 1:M relationship requires that when the standard database design algorithms are used to transform the entity-relationship model into the relational database schema (Figure 8), this relationship's attributes will be combined with the attributes of the MAINTENANCE TRANSACTION REPORT and FUEL/FLIGHT HOUR REPORT entities. However, in this case there are no attributes associated with the relationships, so there is no requirement to merge attributes into these tables.

- Squadron/CVW - This relationship is used once. It links CVW and SQUADRON entities. The relationship is of degree two and requires total participation of the CVW entity. Being a M:N relationship, this relation will become a relational table with its own unique attributes. The relation identifies specific SQUADRONS which are assigned to specific CVWs.

Figure 8. AFAST II Relational Model

C. AFAST II DATABASE IMPLEMENTATION

This section discusses the criteria utilized to select the database management system (DBMS), the general purpose software system that facilitates the process of defining, constructing and manipulating databases for various applications [Elmasri94]. It was known from the outset that the ultimate DBMS to be employed by the AFAST II operational system would be ORACLE7®; AIRPAC had specified that, based on personnel, training and licensing considerations, an ORACLE database application would be preferred.

AFAST II prototype development was not required to utilize ORACLE, and given this latitude we elected from the outset to use commercially available software tools which would allow us to pursue a proof of concept design, while at the same time minimize developmental costs. Guiding our choice of software tools were the following criteria:

- Software functionality should be similar to that of Oracle. Experience gained and lessons learned during prototype development should be readily transferable and applicable to the operational database development and implementation.

- Software should posses capability to support a large multiuser database; it must be able to operate in a client/server environment.

- DBMS should have the capability to control redundancy so as to prohibit inconsistencies among the data. Controls should be automatically executed and enforced.

- DBMS should provide a security and authorization subsystem, restricting user's access to data.

- DBMS must have the capability to represent complexities among the data as well as retrieve and update related data easily and efficiently.

- DBMS must provide capabilities for defining and enforcing integrity and referential integrity constraints. While integrity constraints would be identified

by the designer during database design, the DBMS would be required to enforce these constraints automatically during system runtime.

- Finally, the DBMS must provide facilities for recovering from hardware or software failures.

After testing and analysis of three similarly priced software products the Borland® Interbase® Server was selected for the development and implementation of the AFAST II prototype. All three products tested, in varying degrees, satisfactorily emulated the procedures and functionality of ORACLE7®. The deciding factor was the software's ability to handle complex relations. Borland® Interbase® allows the designer to create M:N relationships and execute, without complication, third degree relationships.

Borland® Interbase® provided another significant advantage in that it was the database utilized by Borland® Delphi™, the application package which was chosen to design the graphical user interface. In general, the software provides a tightly integrated set of client/server tools necessary for building, testing and deploying a client application earmarked to interface with a remote server. Features which made Interbase® our choice were:

- Use of stored procedures and triggers
- Effective referential integrity procedures
- Immediate recovery from system failure
- Significant security and access properties
- Scalability

The last feature, scalability, is key. AIRPAC has expressed a desire to integrate the AFAST II prototype into their architecture as soon as feasible with the expressed intent of validating data output by the current AFAST system against the AFAST II (prototype) application, with AFAST II being the accepted standard of accuracy.

IV. AFAST II USER INTERFACE DESIGN

A. INTRODUCTION

This chapter focuses on the second part of the Kroenke and Dolan design phase: the development of the application design. The primary goal of the application design is to create a sensible, comprehensible, memorable and convenient semantic organization relevant to the user's tasks [Shneiderman93]. Three distinct application formats are available for developing and implementing the design:

- command languages
- direct manipulation
- menu selection

B. COMMAND LANGUAGES AND COMMAND LINE INTERFACES

Command languages, which originated with operating system commands, are distinguishable by their immediacy and by their impact on devices or information. Users issue a command and watch for effect. If the result is correct the next command is entered, if not, some other strategy (i.e., reissue command, error recovery) is adopted. "In general commands are brief, transitory and produce an immediate result on some object of interest [Shnedierman93]."

A command line format requires the user to enter a command, or series of commands, when prompted by the operating system. These commands must be understood and retained by the user, who oftentimes is required to memorize hundreds of

commands, flags, syntactical formats and syntax variations. Command languages and command line interfaces are of great utility when the users are knowledgeable and quite comfortable with the task and the user interface application. Within the scope and framework of AFAST II it is an invalid assumption to suggest that even the most frequent user would be knowledgeable and comfortable with the database query languages which would be used as the basis for the command language interface. Therefore other application formats must be considered.

C. GRAPHICAL USER INTERFACES

The benefit of graphical user interfaces (GUIs), when compared to command languages and command line interfaces, is that the representation of the *miniworld* may be closer to the more familiar structure in which the user functions. Making use of the GUI presents to the designer opportunities, in the case of AFAST II, to better describe query formulation and relationships through the use of proximity, containment or color coding. McCormick emphasizes the potential of GUIs when he states: "Visualization is a method of computing. It transforms the symbolic...enabling [users] to observe their simulations and computations. Visualization offers a method for seeing the unseen. It...fosters profound and unexpected insights [McCormick87]."

1. Direct Manipulation

A direct manipulation interface is one where users perform action directly on objects. This is in contrast to interfaces, such as the command line interface, where the

users indirectly specify actions, parameters and objects through language. A direct manipulation interface has the following characteristics [Mayhew92]:

- Continuous representation of objects.
- Physical actions or label button presses in place of command languages.
- Rapid incremental reversible operations with immediate, visible results.

Direct manipulation has distinct advantages over other dialog styles. These include [Shneiderman93, Mayhew92]:

- Easy to learn and remember - Once a user becomes aware of the analogies to manipulate objects in space and fluent in the uses of the pointing device, then even functionally complex applications become relatively easy to learn.
- What you see is what you get (WYSIWYG) - Direct manipulation interfaces are truthful and, as the name implies, direct, permitting the user to concentrate on the task rather than on the computer system syntax. Interaction is more comfortable for the user, promoting greater efficiency and less errors.
- Flexible. Easily reversible actions - If the user did not intend to select an object, one quick click of the mouse deselects or reverses the unplanned action. Users experience less anxiety because the system is comprehensible and so easily reversible.
- Provides instant visual feedback - When the mouse button is depressed to execute an action, the results are shown immediately on the screen. When the application involves moving around the screen, users get direct, visual feedback while the object is being moved.
- Exploits human use of visual-spatial cues - Users rely on visual information and are usually faster in processing a picture or diagram than a verbal description of the same object. Direct manipulation exploits this human ability.

The key to the successful development and implementation of a direct manipulation interface is the design of a stable and true representation of the miniworld.

When the interface is constructed well "the user is able to apply intellect directly to the task, the tool itself seems to disappear" [Rutkowski82].

2. Menu Selection

Menu selection is appealing because it can significantly reduce training and memorization requirements placed upon the user. This application format is effective when users have little computer expertise, use the system intermittently, are unfamiliar with the application terminology or need assistance in structuring their decision making process. Contrasted with the command language format, the menu format is much simpler for a new user to learn. "The simple menu provides the user with an overall context and is less error prone than the command line format" [Pressman92].

Design issues for menu systems can be divided into five areas. These include [Mayhew92]:

- menu structure
- menu choice ordering
- menu choice selection
- menu invocation
- menu navigation

a. Menu structure

When a collection of items grows and becomes too difficult to maintain under intellectual control, people form categories of similar items. Some collections can be partitioned easily into mutually exclusive groups with distinctive identifiers

[Norman91]. This technique, known as multidimensional scaling, produces sets of items that are representative of how users view relationships between items, and is often considered a logical way to categorize items in a menu system.

b. Menu choice ordering

Once a menu structure is resolved and the items that will appear on each menu screen have been determined, the designer is still confronted with the question of ordering for the items on each menu screen. Ordering schemes include:

- Natural sequence (i.e., time, numeric, physical properties such as weight)
- Alphabetic sequence
- Most frequently used first
- Most important first

Mayhew recommends: "Order menu choice labels according to convention, frequency of use, order of use, categorical or functional groups and/or alphabetical order. If no other ordering scheme lends itself well to the menu choices, then alphabetic is better than random ordering, especially for high-frequency...users" [Mayhew92].

c. Menu choice selection

Numerous selection tools are available for implementation by the designer. The most common and accepted standard for manipulating menus of applications which incorporate graphical user interfaces is the mouse. Alternate selection means, such as keyboard selection codes (i.e., tab and arrow keys), should

also be used in the application; frequent users may prefer entering commands via the keyboard and are able to execute actions faster than by pointing with the mouse.

d. Menu invocation

In most menu systems, the top level menu is permanently visible in some reserved area of the screen. Newer menu systems have pop-up, embedded or user-invoked menus, in which the user must initiate some action (e.g., click the right mouse button) in order to display the menu. The use of pop-up and embedded menus allows designers to provide detailed information to the user without sacrificing screen real estate since the menu exists only in a temporal state. The major drawback to the use of menu invocation is the requirement placed upon the user to know what functions are available through the menus, and remembering how to invoke the correct menu.

e. Menu Navigation

Enabling the user to comfortably transit through the application is accomplished by a number of menu structural design considerations. These include:

- Conventional and standardized functions available on all menu screens.
- Consistent characterization and employment of functions on all menu screens (e.g., return always returns the user to one level above the present location).
- Use of labels and place markers as navigational aids in menu systems.
- Employment and consistent utilization of menu types (pop-up, embedded, cascading), dialog boxes and dialog box controls (buttons, form fillin, fields, scroll bars).

- Backward navigation; it should always be possible to return quickly from any menu back to the main menu, as well as back up one level at a time.

D. APPLICATION DESIGN AND IMPLEMENTATION

1. Development Process

The user interface is one of the most problematic components of a database system. Interface design is a matter of compromise and trade-off. The designers want robust functionality but require a simple, unclouded interface. An overall objective is consistency across all aspects of the interface, but this must be tempered against the requirements of optimizing individual operations. The designers strive for ease of use and ease of learning, employing various methods and techniques to effectively manage the user interface design and help make good design decisions for a given product based on its specific set of end users.

The methods and techniques consist of design tasks applied in a specific order at specific milestones in the database development. These tasks are:

- Scoping
- Functional Specification
- Design
- Development
- Testing/Implementation

Scoping and Functional Specifications are summarized in the preceding chapters and defined in the following documents:

- AFAST System Design Document dtd November 1995
- AFAST Functional Description dtd November 1994
- NPS AFAST Research Proposal dtd March 1996

Further testing and verification of the database queries of AFAST II is to be conducted following the completion of this document. The design of the AFAST II database and application interface were our prime objectives for this thesis.

There are three principal processes associated with AFAST II:

- Casual user interface; reading the database (Figure 9).
- Expert user interface; reading/writing to the database.
- Graphics generation.

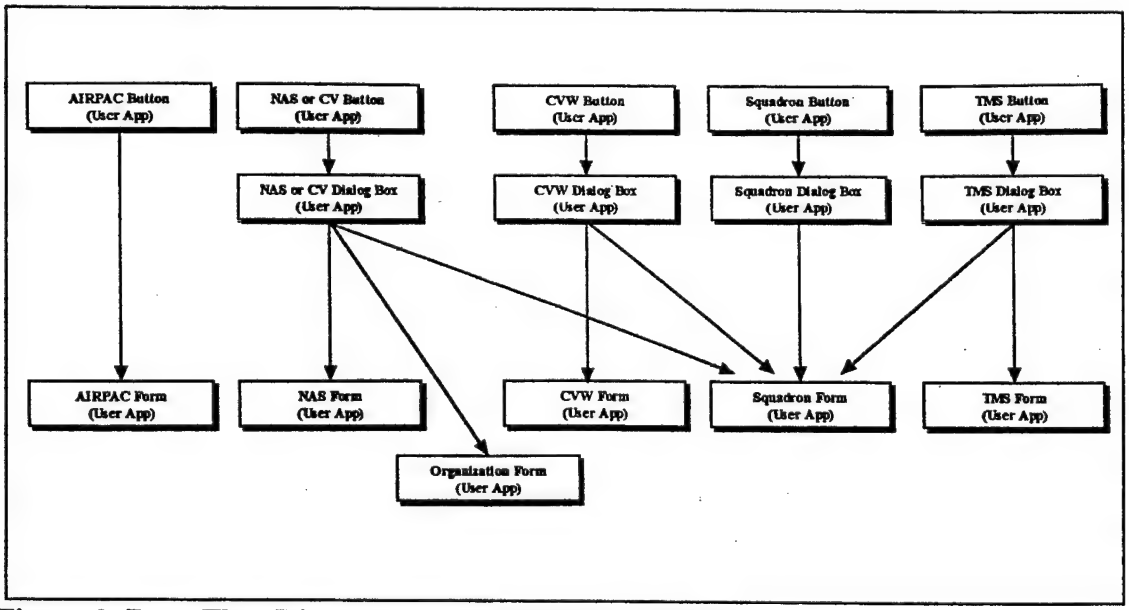


Figure 9. Data Flow Diagram for Casual User Interface

For AFAST II the translation of processes into applications is one-to-one, with each application defined by its own specifications, detailing how entities, attributes, relationships and queries are to be presented. Once the applications are determined and

defined, the search for a software tool which will enable the development of the application can begin. As stated in Chapter III, a number of factors played critical roles in the choice of the application development package. Cost, functionality and compatibility with the current AIRPAC architecture were those conditions which directly influenced the decision to choose Borland™ Delphi™ as the software development application.

Delphi's integrated development environment provides a number of tools for creating user interfaces. Using Delphi, we created the visual elements quickly, and were able to easily conform to the guidelines associated with the development of menus and direct manipulation interfaces. Additionally, the product enabled us to create a client application that was self contained and totally separate from the database. This will allow a smooth transition from Interbase to Oracle without making any changes to the application interface.

The scope of this document details only the design and development of the casual user application. The remaining applications design efforts will be chronicled in follow-on documentation [NPS96].

2. Menu Selection Methodology

Central to the application development is the use of standard window functions designed to assist the user in navigating through the interface. These functions are:

- Menu bar
- Command key equivalent (shortcut keys) to menu bar commands
- Toolbar

Most casual users are familiar and comfortable with menu bar and shortcut keys as both are greatly utilized in modern windows applications. The incorporation of a Toolbar into the application provides users with an alternate, more efficient means to navigate to the more frequently used queries and operations.

a. Menu bar and Command key equivalents

The menu bar is permanently located at the top of all AFAST II windows. It provides the user ready access to the following major menus (Figure 10):

- File
- Edit
- Forms
- Options
- Window
- Help

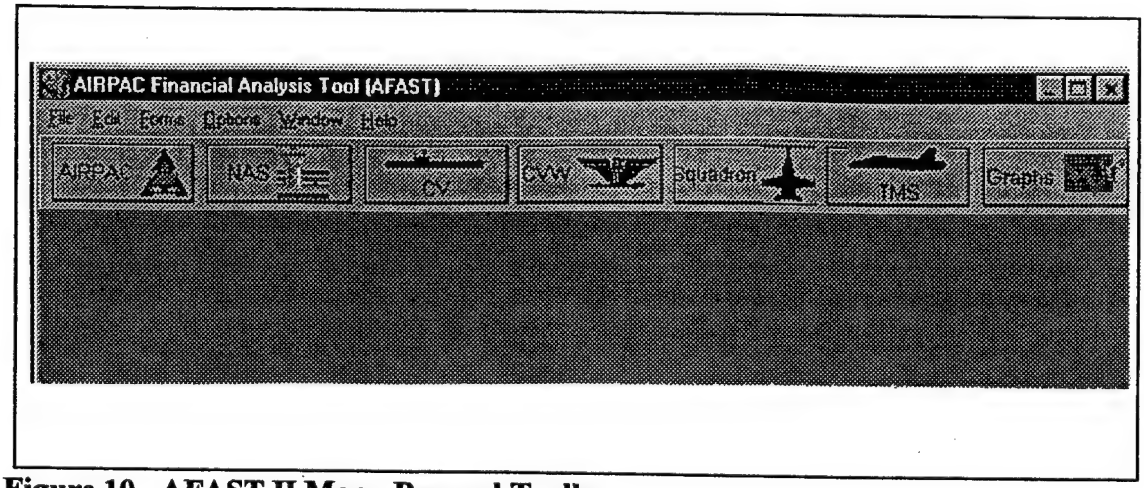


Figure 10. AFAST II Menu Bar and Toolbar

When the File menu option is selected, a cascading menu is generated which provides the user with the following options: close the active child window, print the

active child window, access the Printer setup window and enter the desired printer configuration, or exit the application. The File menu option is regularly utilized in most commercial windows applications. The use of this option by AFAST II presents the user with familiar and consistent commands which are readily understood and easily invoked.

Selecting the Edit menu option generates a cascading menu which provides the user with the capability to copy the active child window in bitmap format to the clipboard. The Edit menu option is also found in most commercial windows applications, and its employment in AFAST II provides the same benefits as does the File menu option.

The third menu option available to the user is Forms. When it is selected a cascading menu is brought forward, displaying various major command categories that are available for display. These are: AIRPAC, NAS, CV, CVW, Squadron, and TMS. This menu may be utilized by the user when the toolbar is hidden or if the user prefers to use shortcut keys. Selection of a command category, such as Squadron, performs the same task as pressing the button Squadron on the toolbar, thus generating a dialog box. Also included in the Forms menu option is the category Graphs. This will allow the user to navigate to the graphics application.

Selection of the Options menu option permits the user to modify application default conditions. When selected a cascading window is opened with four items: Period, Late Report, Exceptions, and Toolbar. Period allows the user to modify the time frame used in querying the database. Late Report identifies all commands delinquent in submitting ASKIT or NACA reports. Exceptions allows the user to set cost variance thresholds for specific type commands (i.e., NAS's). The last option, Toolbar, enables the user to display or hide the toolbar.

The Window menu option provides the following functions: Tile, Cascade, Arrange Icons, Close All. AFAST II provides significant improvement over AFAST through the use of a multiple document interface (MDI). Because AFAST is a single document interface (SDI), it only allows the user to display one set of queried data, where AFAST II generates a new child window upon each user query. AFAST II provides the user the capability to compare side-by-side queries or revisit an old query by simply arranging windows.

The Windows menu option transfers to the user the capability to manage the child windows. Tile divides the screen into equal parts, giving each window part of the visible screen. If three windows are open and tile is selected, each window will be given one-third of the screen, with the screen being partitioned horizontally. Cascade places all child windows in a cascading fashion but in such a manner that the window legend is visible and that a window can be selected by the mouse to be brought to the forefront of the screen. Finally, Close All closes all open child windows.

The last menu option is Help. Selection of this item reveals a cascade menu with three functions: Index, Search and About AFAST. When implemented they are designed to provide access to form fill-in menus which will enable the user to query the help file and receive assistance as required. About AFAST currently provides the user with version information and designer data.

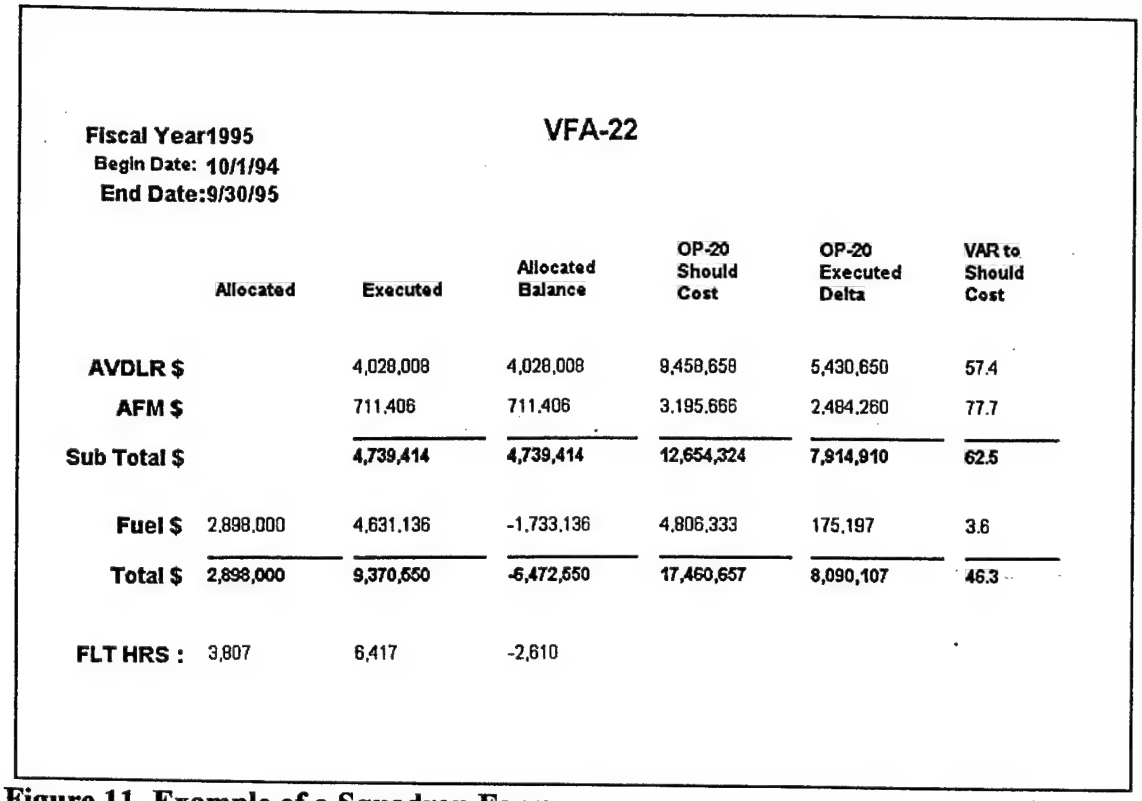
b. Toolbar

The purpose of the toolbar is to provide the user with a rapid means to access the various major command categories (AIRPAC, NAS, CV, CVW, Squadron,

TMS) and graphics application. The use of icons offers a natural representation of the task objects and actions (i.e., direct manipulation), while at the same time presenting an alternative interface for the high frequency, expert user. Individual icons present images commonly associated with the various categories (e.g., a carrier silhouette represents CV's) with associated text which amplifies the icon image. The buttons are given color and three dimensional shading to make them visually and conceptually distinct from each other and the remainder of the interface. The toolbar buttons perform the same functions associated with their Forms menu option counterparts. When selected each button will navigate the user through a selection process which yields specific data for the major command type selected.

3. Child Window Form Descriptions

The general windows designs used to display data (Appendix C) are guided by the AIRPAC desire to have AFAST II resemble, as much as feasible, the current AFAST system. Figure 11 displays a typical windows data display for a squadron. The tabular format is standard whether the data being displayed represents AIRPAC, an NAS, a CV, a CVW, a Squadron, or a TMS. Specific columns and rows may remain empty or be deleted from the form entirely if the data for the command is not applicable (e.g., CV's and NAS's are not allocated flight hours or fuel dollars, these data fields are blank on their respective windows). Descriptions of each window are provided below. The data flow diagram, which delineates the path which must be navigated to access each window, is provided in Appendix C.



Fiscal Year1995
Begin Date: 10/1/94
End Date:9/30/95

VFA-22

	Allocated	Executed	Allocated Balance	OP-20 Should Cost	OP-20 Executed Delta	VAR to Should Cost
AVDLR $		4,028,008	4,028,008	9,458,658	5,430,650	57.4
AFM $		711,406	711,406	3,195,666	2,484,260	77.7
Sub Total $		4,739,414	4,739,414	12,654,324	7,914,910	62.5
Fuel $	2,898,000	4,631,136	-1,733,136	4,806,333	175,197	3.6
Total $	2,898,000	9,370,550	-6,472,550	17,460,657	8,090,107	46.3
FLT HRS :	3,807	6,417	-2,610			

Figure 11. Example of a Squadron Form

a. AIRPAC Summary Form

This window is accessed via the Forms menu bar option or the AIRPAC button on the toolbar. All fields in the AIRPAC summary table are applicable and represent AIRPAC subordinate commands total allocation and executed costs for all. Improvements made to this window in the AFAST II application are:

- rearranging columns to better reflect total maintenance costs.
- more distinct date legend, enabling user to quickly determine period of report.

b. NAS Summary Form

This window is accessed via the Forms menu bar option or the NAS button on the toolbar. When NAS is selected a dialog box is displayed in the center of the screen. From the dialog box (discussed below) the user can select a particular NAS. All fields are applicable with the exception of those in the Fuel $ row (only squadrons are allocated fuel dollars). The data in this table represents the costs for total maintenance rendered by this specific air station and the number of flight hours that have been flown at this NAS by tenant or deployed squadrons. Improvements made to this window in the AFAST II application are:

- dialog box allows specific NAS selection.
- rearranging columns to better reflect total maintenance costs.
- more distinct date legend, enabling user to quickly determine period of report.

c. CV Summary Form

This window is accessed via the Forms menu bar option or the CV button on the toolbar. When CV is selected a dialog box is displayed in the center of the screen. From the dialog box the user can select a particular CV. All fields are applicable with the exception of those in the Fuel $ row (only squadrons are allocated fuel dollars). The data in this table represents the total costs for maintenance rendered by this specific carrier and the number of flight hours that have been flown at this CV by attached squadrons. Improvements made to this window in the AFAST II application are:

- dialog box allows specific CV selection.

- rearranging columns to better reflect total maintenance costs.
- more distinct date legend, enabling user to quickly determine period of report.

d. CVW Summary Form

The window is accessed via the Forms menu bar option or the CVW button on the toolbar. When CVW is selected a dialog box is displayed in the center of the screen. From the dialog box the user can select a particular CVW. All fields are applicable with the exception of AVDLR $ and AFM $ Allocated (only NAS's and CV's are allocated maintenance dollars). The data in this table represents the total flight hour and fuel costs generated by all squadrons assigned to this carrier air wing. Improvements made to this window in the AFAST II application are:

- dialog box allows specific CVW selection.
- rearranging columns to better reflect total maintenance costs.
- more distinct date legend, enabling user to quickly determine period of report.

e. Squadron Form

The window is accessed via the Forms menu bar option or the Squadron button located on the toolbar. When Squadron is selected a dialog box is displayed in the center of the screen. From the dialog box the user can select a particular Squadron via the aircraft mission type. All fields are applicable with the exception of AVDLR $ and AFM $ Allocated (only NAS's and CV's are allocated maintenance dollars). The data in this table represents the total flight hour and fuel costs generated by the squadron both home and deployed. Improvements made to this window in the AFAST II application are:

- dialog box allows specific Squadron selection.
- rearranging columns to better reflect total maintenance costs.
- more distinct date legend, enabling user to quickly determine period of report.
- use of tabbed notebook to access detailed squadron information.

Each of the tabs of the notebook contains detailed information for each TMS the particular squadron flies. When a particular TMS tab is selected (Figure 12) the maintenance and fuel/flight hour data specific to that TMS is brought to the forefront of the child window. This notebook concept allows the user to visually ascertain the existence of all the TMS's that a squadron flies. The Squadron window will have at least one TMS window, and may have many more. The number of notebook pages is determined by the number of TMS's assigned to the squadron and resident in the database.

VFA-22
FA-18C

Fiscal Year1995
Begin Date10/1/94
End Date:9/30/95

		CNAP Execution	OP-20 Should Cost		CNAP Execution	OP-20 Should Cost
AVDLR		4.028.008	9.458.658	FUEL	4.631.136	4.806.333
Pwr Plants	130,368					
Avionics	3.002.952			FLT HRS	6,417	
Air Frames	894,688					
Other	0					
Misc TEC	0					
Overhead	0					
AFM		711,406	3,195,666	COST PER HOUR		
Pwr Plants	6,089					
Avionics	99,249			FUEL	722	749
Air Frames	130,269			AVDLR	628	1,474
Other	475,799			AFM	111	498
Misc TEC	0					
Overhead	0					

Figure 12. Example of a Detailed Squadron Form

The use of a tabbed notebook is yet another example of direct manipulation principles integrated into the AFAST II design. Actions are rapid, incremental and reversible, and can be performed with seemingly transparent physical actions instead of complex queries. The results of the action (tab selection) are immediate and predictable.

f. TMS Form

This window is accessed via the Forms menu bar option or the TMS button on the toolbar. When TMS is selected a dialog box is displayed in the center of the screen. From the dialog box the user can select a particular TMS. The window design (Figure 13) is similar to the basic format utilized for the preceding windows, however there exists two notable differences. The first variance is all mission types, FRS, Support, TACAIR, Total, (subsets of the TMS), are displayed on the first page of the notebook. This format is a holdover from the original AFAST design and is intended to allow the user to view summary data associated with the TMS and its major mission categories. The second variance is that the table has been modified by deleting all data fields which are not relevant, with the most significant benefit being an increase in available screen real estate and the opportunity to create a much cleaner, uncluttered and user friendly table. Consistency with AFAST is adhered to by using the identical row and column titles as well as maintaining the same relative ordering of the remaining columns. A TMS has a tab for each mission type (Figure 13) that it is assigned. When the tab is selected specific maintenance and fuel/flight hour data is displayed for the TMS's mission type, as well as a listing of squadrons which have been allocated funds to fly that mission with the specified TMS.

FA-18C

Fiscal Year:1995
Begin Date:10/1/94
End Date: 9/30/95

		CNAP Execution	OP-20 Should	VAR to Should Cost
FRS	AVDLR $		5,294,160	100
	AFM $		2,124,200	100
	FUEL $	3,251,437	3,378,295	4
	FLT HRS	4,085		
SUPPORT	AVDLR $	0	0	
	AFM $	0	0	
	FUEL $	0	0	
	FLT HRS	0		
TACAIR	AVDLR $	7,451,160	51,648,960	85.6
	AFM $	1,829,225	17,449,820	89.5
	FUEL $	26,153,868	26,244,960	0.3
	FLT HRS	35,040		
TOTAL	AVDLR $	7,451,160	56,943,120	86.9
	AFM $	1,829,225	19,574,120	90.7
	FUEL $	29,405,405	29,623,255	0.7
	FLT HRS	39,125		

Figure 13. Example of a TMS Form

4. Linear sequences and dialog boxes

A succession of interdependent dialog boxes is used to guide the user through a series of choices to display the requested data. The dialog boxes are consistently designed and act in a similar manner. Movement through menus and dialog boxes is a linear sequence, a designed methodology used to guide the user through a decision making process by presenting one decision at a time [Shneiderman93]. This is the philosophy behind the design and implementation of the dialog boxes used in AFAST II to enable the user to display data for specific NAS's, CV's, CVW's, Squadron's and TMS's.

Most of the major command category buttons on the toolbar are associated with a dialog box. In Figure 14, the NAS dialog box is presented to the user who is now able to immediately determine:

- The NAS's stored in the database.
- Last selected NAS (highlighted by color bar and carrot).
- Squadrons home-ported or deployed to the NAS.
- Organizations which have spent maintenance funds allocated to the NAS.

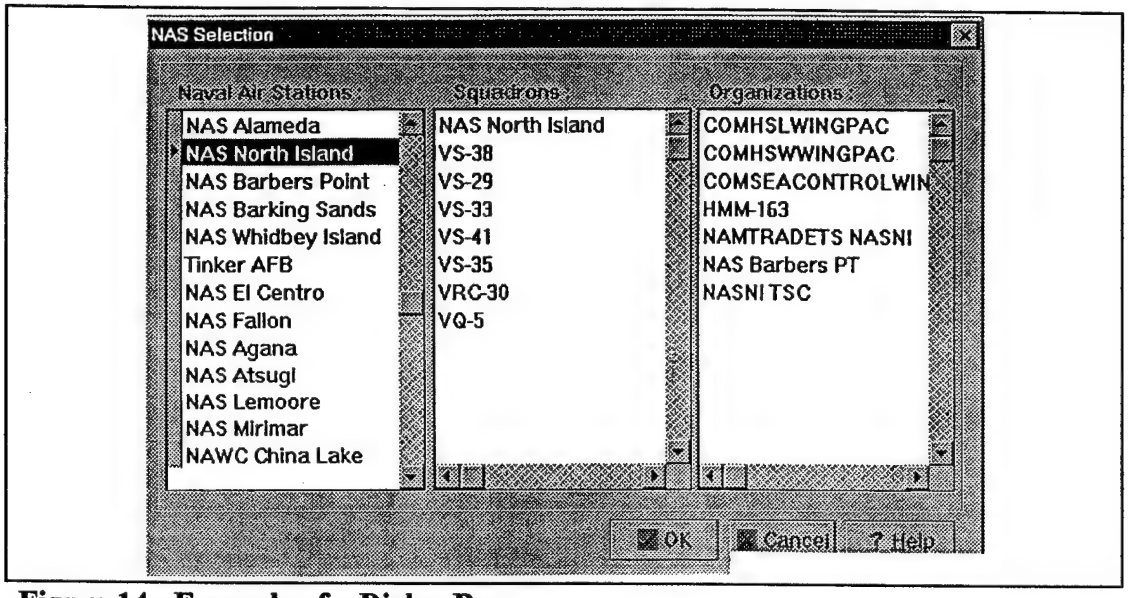


Figure 14. Example of a Dialog Box

The user then has the option to: select an NAS and display the NAS; select an NAS, then a squadron, and display the squadron data; select an NAS, then an organization, and display the organizational data. Should the user decide that he no longer wishes to view NAS data he may close the dialog box using the Cancel button. If the user is unfamiliar with the window methodology, as might be in the case of a naïve user, depressing the Help button will call the specific Help documents.

The dialog boxes are designed to rapidly move the user through the prefatory and mundane selection process and place him/her at their intended location as quickly as possible. Additionally, simplistic and logical error recovery techniques are integrated

into the methodology employing the *What You See Is What You Get* philosophy; if the user is not satisfied with his selection choice he merely clicks on another selection or exits from the dialog box. The dialog boxes offer a superior improvement over what is presented to the user in the original AFAST interface, which are:

- One list box (e.g., NAS selection) per dialog box.
- Sub-menus where only one squadron or one organization could be viewed at any one instance.
- No convenient error recovery.
- No indications provided to user as to his location in the menu hierarchy.

The use of dialog boxes to facilitate linear progression through the application allows the user to focus more effectively on the task semantics, or the functional task, being less concerned with system semantics. This is a much more natural, more efficient and less error prone than the original AFAST method.

5. Prototype Performance

A consequential advantage of using Borland® Delphi™ tools for both the database and application development was the ease with which both were coupled to create a database prototype. The prototype was installed on an IBM-compatible personal computer, a pentium, 166Mhz with 32 Megabytes of memory. The purpose of the prototype testing was qualitative in nature, designed to:

- Validate the database - i.e., the entities, attributes, relationships, triggers, stored procedures.
- Validate query logic used in the user application.

- Confirm window features and functions performed as intended when provided sample data from the database.

Quantitative analysis of the prototype was never intended as the differences in performance between the prototype and ultimate hardware configuration are too significant. However, important data was attained in the qualitative analysis of the database and interface performance. This data is presented in Chapter V.

V. CONCLUSIONS AND RECOMMENDATIONS

A. SUMMARY

The objectives of this thesis were to:

- Develop an improved conceptual design of the AFAST database using the enhanced entity-relationship model concepts and procedures.
- Specify the logical design of the improved database.
- Implement AFAST II, a prototype application of the redesigned database.

Enroute to meeting these objectives a thorough technical review of relational database design methodology, human-computer interface design issues, and commercial software database development tools was conducted. The information collected through the course of this effort reinforced the underlying premise in the design of AFAST II, that there is benefit to be gained by:

- Utilizing recognized design methodologies in the conceptual and logical model development.
- Placing the design task and prototype development in the hands of schooled and capable domain experts.

As work progressed it became apparent that the success of AFAST II would not lie in providing optimal solutions or high performance algorithms, rather the application's worth lies in its ability to provide the user with an explicit understanding of the AIRPAC flight hour program. While much has been accomplished, the task is not yet complete. This thesis provides the framework and cornerstone for ongoing work which will achieve

the goal of delivering a management information system to assist AIRPAC in its quest to operate in a more business-like manner.

B. ADDITONAL RESEARCH AND DEVELOPMENT

Further work is necessary to meet the requirement to deliver an operational database to CNAP no later than September 1997. Important and consequential milestones which must be met are delineated below.

1. Development of the Expert User and Graphics interface

Two interfaces remain to be developed. These are:

- Expert User Interface. The database administrator application permitting the user to execute queries which read and write to the database.
- The Graphics generation interface which transforms tabular data into chart and graphics output.

The ease with which the normal user interface was developed and the robustness which the system exhibited during qualitative evaluation compel us to recommend that the Borland® Delphi™ tools continue to be utilized for application design, development and implementation.

2. Definition of AFAST II hardware configuration

Currently all that has been specified for the targeted system is that the database will reside on a Sun/Sparc server, and the clients will consist of an undetermined number of IBM compatible personal computers. Efforts must be initiated to define hardware

specifics, number of clients and any hardware modifications which must be implemented to the AIRPAC LAN to incorporate AFAST II.

3. Definition of AFAST II software configuration

The specifications for the software are more definitive than the hardware, yet more work must be done to ensure that the software design is compatible with the hardware configuration. Presently it is envisioned that all interface applications will be created using Borland® Delphi™ tools and will reside on the client side. The database will be developed using Oracle and will be resident on the server. Clients and Server will communicate via the AIRPAC LAN, an Ethernet which uses Novell netware. As was the case for the hardware configuration, the software specifics must be identified and prototyped to ensure the desired configuration is a feasible configuration.

4. Modifications to reporting procedures

The ASKIT and NACA reports are data inputs which are extracted from other programs and "shaped to fit" into AFAST II. NPS has frequently encouraged AIRPAC to review the reporting procedures associated with these reports, and revise them to better enable and expand the capabilities of AFAST II. It is agreed by all involved in the AFAST process that these reports will eventually undergo modification. We recommend that these changes be done early on in order to minimize the rewriting of stored procedures and triggers.

5. Validation of the AFAST II database algorithms

Limited testing using a small database was conducted to verify that AFAST II algorithms performed as designed. More thorough testing to include error checking and exception handling must be conducted prior to system delivery. For testing purposes, the database should be populated with current fiscal year data, with query results compared to those results provided by the current AFAST system. Deviations between the two systems should be accounted for (i.e., round-off error, updated algorithm used) and documented.

6. Validation of AFAST II application interfaces

It is recommended that thorough test scenarios be generated to evaluate the prototype performance. Test subjects, preferably naval aviators, can be drawn from the NPS population and observed as they interface with the application. Critical to note will be their actions, perceptions and approach to problem solving. A qualitative methodology is recommended, as it can:

- Better help to identify areas where users are having problems.
- Provide information necessary to improve the design and development process.

7. Expansion of the Help function

At the present time the Help function, accessed via the main menu bar, is not implemented. Though not required by AIRPAC we strongly recommend Help procedures delineating how to accomplish specific queries (i.e., display Squadron data for a Squadron assigned at a specific NAS) be implemented into the first deliverable application.

Additional Help features which should be incorporated into follow-on deliveries include definitions of terms and explanations of algorithms used to perform database queries.

APPENDIX A. DATABASE SOUCE CODE

This appendix contains the source code for the definition of the AFAST II database. Specifically, the Interbase scripts that were used to input the metadata are presented.

A. DOMAIN DEFINITION

```
CONNECT "d:\thesis\database\afast_db"
USER"sqldba" PASSWORD "masterkey";

CREATE DOMAIN UIC AS CHAR(6)
        NOT NULL
        CHECK ((VALUE > '999999') AND    /* first char a letter */
        (VALUE STARTING WITH 'R') OR
        (VALUE STARTING WITH'N'));

CREATE DOMAIN NAME AS VARCHAR(25)
        DEFAULT 'NONE'
        NOT NULL;

CREATE DOMAIN MILITARY_SERVICE AS VARCHAR(4)
    DEFAULT 'USN'
        NOT NULL
        CHECK (VALUE IN ('USN','USMC'));

CREATE DOMAIN MISSION_TYPE AS CHAR(1)
        NOT NULL
        CHECK (VALUE IN ('T', 'S', 'R'));

CREATE DOMAIN TYPE_NAME AS VARCHAR(5)
        NOT NULL
        CHECK (VALUE IN ('VFA','VA','VF','VS','VAQ','VAW','VX','VMFA','VRC','VQ','HS',
'HC','HM','HSL','VP'));

CREATE DOMAIN NAS_OR_CV AS CHAR(1)
        DEFAULT 'F'
        NOT NULL
        CHECK (VALUE IN ('T', 'F'));

CREATE DOMAIN TMS_NOMEN AS VARCHAR(7)
        NOT NULL;

CREATE DOMAIN YEAR AS SMALLINT
        DEFAULT '96'
        NOT NULL
```

```
        CHECK (VALUE >= 95);

CREATE DOMAIN TEC_TYPE AS CHAR(4)
        NOT NULL
        CHECK (VALUE STARTING WITH 'A');

CREATE DOMAIN TMS_OP20_COST AS SMALLINT
        NOT NULL
        CHECK (VALUE >= 0);

CREATE DOMAIN START_DATE AS DATE
        NOT NULL
        CHECK (VALUE > 9-30-94);

CREATE DOMAIN END_DATE AS DATE
        NOT NULL;

CREATE DOMAIN FLT_HRS AS INTEGER
        DEFAULT 0
        NOT NULL
        CHECK (VALUE < 9999);

CREATE DOMAIN EXEC_COSTS AS DECIMAL (10,2)
        DEFAULT 0.00
    NOT NULL;

CREATE DOMAIN MAINT_TYPE AS VARCHAR(5)
        NOT NULL
        CHECK (VALUE IN ('AVDLR', 'AFM'));

CREATE DOMAIN BRANCH_TYPE AS VARCHAR(10)
        NOT NULL
        CHECK (VALUE IN ('PWR_PLANTS', 'AVIONICS', 'AIRFRAMES','OTHER', 'MISC_TEC',
'OVERHEAD'));

CREATE DOMAIN MAINT_COST_TYPE AS CHAR(1)
        NOT NULL
        CHECK (VALUE IN ('D','O'));

CREATE DOMAIN OP20_FLT_HRS AS INTEGER
        NOT NULL
        CHECK (VALUE > 0 AND VALUE < 500000);

CREATE DOMAIN OP20_FUNDS AS DECIMAL(12,2)
        NOT NULL
        CHECK (VALUE > 0);

CREATE DOMAIN CVW_END_DATE AS DATE;

CREATE DOMAIN NACA_UIC AS CHAR(6)
        CHECK (((VALUE > '999999') AND
        ((VALUE STARTING WITH 'R') OR
        (VALUE STARTING WITH'N')) OR (VALUE IS NULL)));
```

B. TABLE DEFINITION

```
CONNECT "d:\thesis\database\afast_db"
USER"sqldba" PASSWORD "masterkey";

Create Table Squadron
        (Squadron_UIC                   UIC,
        Squadron_Name                   NAME,
        Service                         MILITARY_SERVICE,
        Mission                 MISSION_TYPE,
        Type                            TYPE_NAME,
        Primary Key(Squadron_UIC));
COMMIT;

Create Table NAS_CV
        (NAS_CV_UIC                     UIC,
        NAS_CV_Name                     NAME,
        CV                              NAS_OR_CV,
        Primary Key (NAS_CV_UIC));
COMMIT;

Create Table Deployed
        (Squadron_UIC                   UIC,
        NAS_CV_UIC                      UIC,
        Remarks                         VARCHAR(50),
        Primary Key (Squadron_UIC, NAS_CV_UIC),
        Foreign Key (Squadron_UIC)
                REFERENCES Squadron (Squadron_UIC),
        Foreign Key (NAS_CV_UIC)
                REFERENCES NAS_CV (NAS_CV_UIC));
COMMIT;

Create Table Organization
        (ORG_Name                       NAME,
        Remarks                         VARCHAR(50),
        Primary Key (ORG_Name));
COMMIT;

Create Table Related_to
        (Org_Name                       NAME,
        NAS_CV_UIC                      UIC,
        Primary Key (NAS_CV_UIC, ORG_Name),
        Foreign Key (NAS_CV_UIC)
                REFERENCES NAS_CV (NAS_CV_UIC),
        Foreign Key (ORG_Name)
                REFERENCES Organization (ORG_Name));
COMMIT;

Create Table TMS
        (ID_TMS                         SMALLINT NOT NULL,
        TMS                             TMS_NOMEN,
        Service                         MILITARY_SERVICE,
        Mission                         MISSION_TYPE,
```

```
        Fiscal_Year                     YEAR,
        TEC                             TEC_TYPE,
        Fuel_Cost_Hour                  TMS_OP20_COST,
        AVDLR_Cost_Hour                 TMS_OP20_COST,
        AFM_Cost_Hour                           TMS_OP20_COST,
        Remarks                         VARCHAR(50),
        Primary Key (ID_TMS));
COMMIT;

Create Table ASKIT
        (ID_ASKIT                       INTEGER NOT NULL,
        Trans_Start_Date                START_DATE,
        Trans_End_Date                  END_DATE,
        TEC                             TEC_TYPE,
        Flt_Hrs                 FLT_HRS,
        Fuel$                   EXEC_COSTS,
        Squadron_UIC                    UIC,
        NAS_CV_UIC                      UIC,
    ID_TMS                          SMALLINT NOT NULL,
        Primary Key (ID_ASKIT),
        Foreign Key (Squadron_UIC)
                REFERENCES Squadron (Squadron_UIC),
        Foreign Key (NAS_CV_UIC)
                REFERENCES NAS_CV (NAS_CV_UIC),
        Foreign Key (ID_TMS)
                REFERENCES TMS (ID_TMS),
    CHECK (Trans_End_Date > Trans_Start_Date));
COMMIT;

Create Table NACA
        (ID_NACA                        INTEGER NOT NULL,
        Squadron_UIC                    NACA_UIC,
        NAS_CV_UIC                      NACA_UIC,
        ID_TMS                                  SMALLINT NOT NULL,
        ORG_Name                        NAME,
        Main_Type                       MAINT_TYPE,
        Branch                          BRANCH_TYPE,
        Cost                            EXEC_COSTS,
        Date_Of_Trans                   START_DATE,
        Cost_Type                       MAINT_COST_TYPE,
        Primary Key (ID_NACA),
        Foreign Key (Squadron_UIC)
                REFERENCES Squadron (Squadron_UIC),
        Foreign Key (NAS_CV_UIC)
                REFERENCES NAS_CV (NAS_CV_UIC),
        Foreign Key (ID_TMS)
                REFERENCES TMS (ID_TMS),
        Foreign Key (ORG_Name)
                REFERENCES Organization (ORG_Name));
COMMIT;
```

```
Create Table FY_Budget
        (Fiscal_Year                    YEAR,
        OP20_Budget_Flt_Hrs             OP20_FLT_HRS,
        OP20_Budget_AVDLR               OP20_FUNDS,
        OP20_Budget_AFM                 OP20_FUNDS,
        OP20_Budget_FUEL                OP20_FUNDS,
        CNAP_Allocated_Flt_Hrs                  OP20_FLT_HRS,
        CNAP_Allocated_AVDLR                    OP20_FUNDS,
        CNAP_Allocated_AFM              OP20_FUNDS,
        CNAP_Allocated_Fuel             OP20_FUNDS,
        Primary key (Fiscal_Year));
COMMIT;

Create Table Sqd_Allocated
        (Fiscal_Year                    YEAR,
        Squadron_UIC                    UIC,
        ID_TMS                                  SMALLINT NOT NULL,
        Hours_Q1                        FLT_HRS,
        Hours_Q2                        FLT_HRS,
        Hours_Q3                        FLT_HRS,
        Hours_Q4                        FLT_HRS,
        Hours_Total                     COMPUTED BY (Hours_Q1 + Hours_Q2 + Hours_Q3 +
Hours_Q4),
        Fuel_Q1                         EXEC_COSTS,
        Fuel_Q2                         EXEC_COSTS,
        Fuel_Q3                         EXEC_COSTS,
        Fuel_Q4                         EXEC_COSTS,
        Fuel_Total                      COMPUTED BY (Fuel_Q1 +  Fuel_Q2 +  Fuel_Q3 +
Fuel_Q4),
        Primary Key (Fiscal_Year, Squadron_UIC, ID_TMS),
        Foreign Key (Squadron_UIC)
                REFERENCES Squadron (Squadron_UIC),
        Foreign Key (ID_TMS)
                REFERENCES TMS (ID_TMS),
        Foreign Key (Fiscal_Year)
                REFERENCES FY_Budget (Fiscal_Year));
COMMIT;

Create Table NAS_CV_Allocated
        (Fiscal_Year                    YEAR,
        NAS_CV_UIC                      UIC,
        AVDLR_Q1                        EXEC_COSTS,
        AVDLR_Q2                        EXEC_COSTS,
        AVDLR_Q3                        EXEC_COSTS,
        AVDLR_Q4                        EXEC_COSTS,
        AVDLR_Total                     COMPUTED BY (AVDLR_Q1 + AVDLR_Q2 +
AVDLR_Q3 + AVDLR_Q4),
        AFM_Q1                                  EXEC_COSTS,
        AFM_Q2                                  EXEC_COSTS,
        AFM_Q3                                  EXEC_COSTS,
        AFM_Q4                                  EXEC_COSTS,
        AFM_Total                       COMPUTED BY (AFM_Q1 +  AFM_Q2 +  AFM_Q3 +
AFM_Q4),
```

```
        Primary Key (Fiscal_Year, NAS_CV_UIC),
        Foreign Key (NAS_CV_UIC)
                REFERENCES NAS_CV (NAS_CV_UIC),
        Foreign Key (Fiscal_Year)
                REFERENCES FY_Budget (Fiscal_Year));
COMMIT;


Create Table CVW
        (CVW_UIC                UIC,
        CVW_Name                NAME,
        Primary Key (CVW_UIC));
COMMIT;

Create Table SQD_CVW
        (SQUADRON_UIC                   UIC,
        CVW_UIC                 UIC,
        CVW_Start_Date                  START_DATE,
        CVW_End_Date            CVW_END_DATE,
        Remarks                 varchar(50),
        Primary Key (Squadron_UIC, CVW_UIC),
        Foreign Key (Squadron_UIC)
                REFERENCES Squadron (Squadron_UIC),
        Foreign Key (CVW_UIC)
                REFERENCES CVW (CVW_UIC),
    CHECK (CVW_End_Date > CVW_START_DATE));
COMMIT;
```

C. TRIGGER DEFINITION

```
CONNECT "d:\thesis\database\afast_db"
USER"sqldba" PASSWORD "masterkey";

CREATE GENERATOR TMS_no_gen;
SET GENERATOR TMS_no_gen to 0;
SET TERM !! ;
CREATE TRIGGER set_TMS_no FOR TMS
BEFORE INSERT AS
BEGIN
   new.ID_TMS = gen_id(TMS_no_gen, 1);
END !!
SET TERM ; !!
COMMIT;

CREATE GENERATOR NACA_no_gen;
SET GENERATOR NACA_no_gen to 0;
SET TERM !! ;
CREATE TRIGGER set_NACA_no FOR NACA
BEFORE INSERT AS
BEGIN
   new.ID_NACA = gen_id(NACA_no_gen, 1);
END !!
SET TERM ; !!
COMMIT;

CREATE GENERATOR ASKIT_no_gen;
SET GENERATOR ASKIT_no_gen to 0;
SET TERM !! ;
CREATE TRIGGER set_ASKIT_no FOR ASKIT
BEFORE INSERT AS
BEGIN
   new.ID_ASKIT = gen_id(ASKIT_no_gen, 1);
END !!
SET TERM ; !!
COMMIT;
```

D. STORED PROCEDURE - AIRPAC

```
CONNECT "d:\thesis\database\afast_db"
USER"sqldba" PASSWORD "masterkey";
SET TERM ^ ;

CREATE PROCEDURE AIRPAC_Summary(FY Integer, BeginDate Date, EndDate Date) RETURNS
(BudgetAVDLR Float, BudgetAFM Float, BudgetFuel Float, BudgetFltHrs Integer,
AllocatedAVDLR Float, AllocatedAFM Float, AllocatedFuel Float, AllocatedFltHrs Integer,
ExecutedAVDLR Float, ExecutedAFM Float, ExecutedFuel Float, ExecutedFltHrs Integer,
ShouldAVDLR Float, ShouldAFM Float, ShouldFuel Float) AS

DECLARE VARIABLE SQD CHAR(6);
DECLARE VARIABLE X   Float;
DECLARE VARIABLE Y   Float;
DECLARE VARIABLE Z   Float;

BEGIN

  SELECT OP20_Budget_AVDLR, OP20_Budget_AFM,OP20_Budget_Fuel,OP20_Budget_Flt_Hrs,
  CNAP_Allocated_AVDLR, CNAP_Allocated_AFM, CNAP_Allocated_Fuel, CNAP_Allocated_Flt_Hrs
  FROM FY_Budget
  WHERE Fiscal_Year = :FY
  INTO :BudgetAVDLR,:BudgetAFM, :BudgetFuel, :BudgetFltHrs, :AllocatedAVDLR, :AllocatedAFM,
   :AllocatedFuel, :AllocatedFltHrs;

  SELECT Sum(Cost)
  FROM NACA NA
  WHERE NA.Main_Type = "AVDLR" and NA.Date_Of_Trans >= :BeginDate and
   NA.Date_Of_Trans <= :EndDate
  INTO :ExecutedAVDLR;

  SELECT Sum(Cost)
  FROM NACA NA
  WHERE NA.Main_Type = "AFM" and NA.Date_Of_Trans >= :BeginDate and
   NA.Date_Of_Trans <= :EndDate
  INTO :ExecutedAFM;

  SELECT Sum(Fuel$), Sum(Flt_Hrs)
  FROM ASKIT A
  WHERE A.Trans_Start_Date >= :BeginDate and A.Trans_End_Date <= :EndDate
  INTO :ExecutedFuel, :ExecutedFltHrs;

  ShouldAVDLR = 0.0;
  ShouldAFM   = 0.0;
  ShouldFuel  = 0.0;

  FOR
    SELECT Squadron_UIC
    FROM Squadron
    INTO :SQD
  DO
    BEGIN
```

```
      SELECT Should_AVDLR, Should_AFM, Should_Fuel
      FROM SHOULD_COST(:SQD, :FY, :BeginDate, :EndDate)
      INTO :X, :Y, :Z;

      ShouldAVDLR =  X + ShouldAVDLR;
      ShouldAFM   =  Y + ShouldAFM;
      ShouldFuel  =  Z + ShouldFuel;

    END
  SUSPEND;
END^
SET TERM ; ^
```

E. STORED PROCEDURE - CVW

```
CONNECT "d:\thesis\database\afast_db"
USER"sqldba" PASSWORD "masterkey";
SET TERM ^ ;

CREATE PROCEDURE CVW_Summary(CVWName Char(25),FY Integer, BeginDate Date, EndDate
Date, Quarter Integer) RETURNS
  (AllocatedFuel Float, AllocatedFltHrs Integer,ExecutedAVDLR Float, ExecutedAFM Float,
  ExecutedFuel Float, ExecutedFltHrs Integer,ShouldAVDLR Float, ShouldAFM Float,
  ShouldFuel Float) AS

DECLARE VARIABLE SQD CHAR(6);
DECLARE VARIABLE X   Float;
DECLARE VARIABLE Y   Float;
DECLARE VARIABLE Z   Float;

BEGIN
  If (:Quarter = 0) then
  BEGIN
    SELECT Sum(Fuel_Total), Sum(Hours_Total)
    FROM CVW C, SQD_CVW SC, SQD_Allocated SA, Squadron S
    WHERE C.CVW_Name = :CVWName and C.CVW_UIC = SC.CVW_UIC and SC.Squadron_UIC =
S.Squadron_UIC
        and S.Squadron_UIC = SA.Squadron_UIC and SA.Fiscal_Year = :FY
    INTO :AllocatedFuel, :AllocatedFltHrs;
  END

  If (:Quarter = 1) then
  BEGIN
    SELECT Sum(Fuel_Q1), Sum(Hours_Q1)
    FROM CVW C, SQD_CVW SC, SQD_Allocated SA, Squadron S
    WHERE C.CVW_Name = :CVWName and C.CVW_UIC = SC.CVW_UIC and SC.Squadron_UIC =
S.Squadron_UIC
        and S.Squadron_UIC = SA.Squadron_UIC and SA.Fiscal_Year = :FY
    INTO :AllocatedFuel, :AllocatedFltHrs;
  END
```

```
If (:Quarter = 2) then
  BEGIN
    SELECT Sum(Fuel_Q2), Sum(Hours_Q2)
    FROM CVW C, SQD_CVW SC, SQD_Allocated SA, Squadron S
    WHERE C.CVW_Name = :CVWName and C.CVW_UIC = SC.CVW_UIC and SC.Squadron_UIC =
S.Squadron_UIC
        and S.Squadron_UIC = SA.Squadron_UIC and SA.Fiscal_Year = :FY
    INTO :AllocatedFuel, :AllocatedFltHrs;
  END

  If (:Quarter = 3) then
  BEGIN
    SELECT Sum(Fuel_Q3), Sum(Hours_Q3)
    FROM CVW C, SQD_CVW SC, SQD_Allocated SA, Squadron S
    WHERE C.CVW_Name = :CVWName and C.CVW_UIC = SC.CVW_UIC and SC.Squadron_UIC =
S.Squadron_UIC
        and S.Squadron_UIC = SA.Squadron_UIC and SA.Fiscal_Year = :FY
    INTO :AllocatedFuel, :AllocatedFltHrs;
  END

  If (:Quarter = 4) then
  BEGIN
    SELECT Sum(Fuel_Q4), Sum(Hours_Q4)
    FROM CVW C, SQD_CVW SC, SQD_Allocated SA, Squadron S
    WHERE C.CVW_Name = :CVWName and C.CVW_UIC = SC.CVW_UIC and SC.Squadron_UIC =
S.Squadron_UIC
        and S.Squadron_UIC = SA.Squadron_UIC and SA.Fiscal_Year = :FY
    INTO :AllocatedFuel, :AllocatedFltHrs;
  END

  SELECT Sum(Cost)
  FROM CVW C, SQD_CVW SC, Squadron S, NACA N
  WHERE C.CVW_Name = :CVWName and C.CVW_UIC = SC.CVW_UIC and SC.Squadron_UIC =
S.Squadron_UIC and
  S.Squadron_UIC = N.Squadron_UIC and N.Main_Type = "AVDLR" and N.Date_Of_Trans >=
:BeginDate
      and N.Date_Of_Trans <= :EndDate
  INTO :ExecutedAVDLR;

  SELECT Sum(Cost)
  FROM CVW C, SQD_CVW SC, Squadron S, NACA N
  WHERE C.CVW_Name = :CVWName and C.CVW_UIC = SC.CVW_UIC and SC.Squadron_UIC =
S.Squadron_UIC and
  S.Squadron_UIC = N.Squadron_UIC and N.Main_Type = "AFM" and N.Date_Of_Trans >= :BeginDate
      and N.Date_Of_Trans <= :EndDate
  INTO :ExecutedAFM;

  SELECT Sum(A.Fuel$), Sum(A.Flt_Hrs)
  FROM CVW C, SQD_CVW SC, Squadron S, ASKIT A
  WHERE C.CVW_Name = :CVWName and C.CVW_UIC = SC.CVW_UIC and SC.Squadron_UIC =
S.Squadron_UIC and
  S.Squadron_UIC = A.Squadron_UIC and A.Trans_Start_Date >= :BeginDate and A.Trans_End_Date
<= :EndDate
  INTO :ExecutedFuel, :ExecutedFltHrs;
```

```
  ShouldAVDLR = 0.0;
  ShouldAFM  = 0.0;
  ShouldFuel  = 0.0;

  FOR
    SELECT S.Squadron_UIC
    FROM CVW C, SQD_CVW SC, Squadron S
    WHERE C.CVW_Name = :CVWName and C.CVW_UIC = SC.CVW_UIC and SC.Squadron_UIC =
S.Squadron_UIC
    INTO :SQD
  DO
    BEGIN
      SELECT Should_AVDLR, Should_AFM, Should_Fuel
      FROM SHOULD_COST(:SQD, :FY, :BeginDate, :EndDate)
      INTO :X, :Y, :Z;

      ShouldAVDLR =  X + ShouldAVDLR;
      ShouldAFM  =  Y + ShouldAFM;
      ShouldFuel  =  Z + ShouldFuel;

    END
SUSPEND;
END^
SET TERM ; ^
```

F. STORED PROCEDURES - NAS_LATE_REPORT, SQD_LATE_REPORT

```
CONNECT "d:\thesis\database\afast_db"
USER"sqldba" PASSWORD "masterkey";
SET TERM ^ ;
CREATE PROCEDURE NASLateReport(LateDate Date) RETURNS (NASNames Char(25)) AS
BEGIN
  SELECT Distinct NAS_CV_Name
  FROM NAS_CV
  WHERE not exists (SELECT NAS_CV_UIC
              From NACA
              WHERE Report_Date >= :LateDate)
  INTO :NASNames;
SUSPEND;
END^
CREATE PROCEDURE SQDLateReport(LateDate Date) RETURNS (SQDNames Char(25)) AS
BEGIN
  SELECT Distinct Squadron_Name
  FROM Squadron
  WHERE NOT EXISTS (SELECT Squadron_UIC
              FROM ASKIT
              WHERE Trans_End_Date >= :LateDate)
  INTO :SQDNames;
SUSPEND;
END^
SET TERM ; ^
```

G. STORED PROCEDURE - NAS_CV SUMMARY

```
CONNECT "d:\thesis\database\afast_db"
USER"sqldba" PASSWORD "masterkey";
SET TERM ^ ;

CREATE PROCEDURE NAS_CV_Summary(NASName Char(25), FY Integer, BeginDate Date,
EndDate Date, Quarter Integer)
RETURNS (AllocatedAVDLR Float, AllocatedAFM Float, AllocatedFuel Float, AllocatedFltHrs Integer,
        ExecutedAVDLR Float, ExecutedAFM Float, ExecutedFuel Float, ExecutedFltHrs Integer,
        ShouldAVDLR Float, ShouldAFM Float, ShouldFuel Float) AS

DECLARE VARIABLE SQD CHAR(6);
DECLARE VARIABLE X   Float;
DECLARE VARIABLE Y   Float;
DECLARE VARIABLE Z   Float;

BEGIN
  if (:Quarter = 0) then
  BEGIN
    SELECT AVDLR_Total, AFM_Total
    FROM NAS_CV_Allocated NA, NAS_CV N
    WHERE N.NAS_CV_Name = :NASName and N.NAS_CV_UIC = NA.NAS_CV_UIC and
NA.Fiscal_Year = :FY
    INTO :AllocatedAVDLR, :AllocatedAFM;

    SELECT Sum(Fuel_Total), Sum(Hours_Total)
    FROM NAS_CV N, Deployed D, SQD_Allocated SA, Squadron S
    WHERE N.NAS_CV_Name = :NASName and N.NAS_CV_UIC = D.NAS_CV_UIC and
D.Squadron_UIC = S.Squadron_UIC
       and S.Squadron_UIC = SA.Squadron_UIC and SA.Fiscal_Year = :FY
    INTO :AllocatedFuel, :AllocatedFltHrs;
  END

  if (:Quarter = 1) then
  BEGIN
    SELECT AVDLR_Q1, AFM_Q1
    FROM NAS_CV_Allocated NA, NAS_CV N
    WHERE N.NAS_CV_Name = :NASName and N.NAS_CV_UIC = NA.NAS_CV_UIC and
NA.Fiscal_Year = :FY
    INTO :AllocatedAVDLR, :AllocatedAFM;

    SELECT Sum(Fuel_Q1), Sum(Hours_Q1)
    FROM NAS_CV N, Deployed D, SQD_Allocated SA, Squadron S
    WHERE N.NAS_CV_Name = :NASName and N.NAS_CV_UIC = D.NAS_CV_UIC and
D.Squadron_UIC = S.Squadron_UIC
       and S.Squadron_UIC = SA.Squadron_UIC and SA.Fiscal_Year = :FY
    INTO :AllocatedFuel, :AllocatedFltHrs;
  END

  if (:Quarter = 2) then
  BEGIN
    SELECT AVDLR_Q2, AFM_Q2
    FROM NAS_CV_Allocated NA, NAS_CV N
```

```
    WHERE N.NAS_CV_Name = :NASName and N.NAS_CV_UIC = NA.NAS_CV_UIC and
NA.Fiscal_Year = :FY
    INTO :AllocatedAVDLR, :AllocatedAFM;

    SELECT Sum(Fuel_Q2), Sum(Hours_Q2)
    FROM NAS_CV N, Deployed D, SQD_Allocated SA, Squadron S
    WHERE N.NAS_CV_Name = :NASName and N.NAS_CV_UIC = D.NAS_CV_UIC and
D.Squadron_UIC = S.Squadron_UIC
        and S.Squadron_UIC = SA.Squadron_UIC and SA.Fiscal_Year = :FY
    INTO :AllocatedFuel, :AllocatedFltHrs;
  END

  if (:Quarter = 3) then
  BEGIN
    SELECT AVDLR_Q3, AFM_Q3
    FROM NAS_CV_Allocated NA, NAS_CV N
    WHERE N.NAS_CV_Name = :NASName and N.NAS_CV_UIC = NA.NAS_CV_UIC and
NA.Fiscal_Year = :FY
    INTO :AllocatedAVDLR, :AllocatedAFM;

    SELECT Sum(Fuel_Q3), Sum(Hours_Q3)
    FROM NAS_CV N, Deployed D, SQD_Allocated SA, Squadron S
    WHERE N.NAS_CV_Name = :NASName and N.NAS_CV_UIC = D.NAS_CV_UIC and
D.Squadron_UIC = S.Squadron_UIC
        and S.Squadron_UIC = SA.Squadron_UIC and SA.Fiscal_Year = :FY
    INTO :AllocatedFuel, :AllocatedFltHrs;
  END

  if (:Quarter = 4) then
  BEGIN
    SELECT AVDLR_Q4, AFM_Q4
    FROM NAS_CV_Allocated NA, NAS_CV N
    WHERE N.NAS_CV_Name = :NASName and N.NAS_CV_UIC = NA.NAS_CV_UIC and
NA.Fiscal_Year = :FY
    INTO :AllocatedAVDLR, :AllocatedAFM;

    SELECT Sum(Fuel_Q4), Sum(Hours_Q4)
    FROM NAS_CV N, Deployed D, SQD_Allocated SA, Squadron S
    WHERE N.NAS_CV_Name = :NASName and N.NAS_CV_UIC = D.NAS_CV_UIC and
D.Squadron_UIC = S.Squadron_UIC
        and S.Squadron_UIC = SA.Squadron_UIC and SA.Fiscal_Year = :FY
    INTO :AllocatedFuel, :AllocatedFltHrs;
  END

  SELECT Sum(Cost)
  FROM NAS_CV N, NACA NA
  WHERE N.NAS_CV_Name = :NASName and N.NAS_CV_UIC = NA.NAS_CV_UIC and
NA.Main_Type = "AVDLR" and NA.Date_Of_Trans >= :BeginDate
      and NA.Date_Of_Trans <= :EndDate
  INTO :ExecutedAVDLR;

  SELECT Sum(Cost)
  FROM NAS_CV N, NACA NA
```

```
  WHERE N.NAS_CV_Name = :NASName and N.NAS_CV_UIC = NA.NAS_CV_UIC and
NA.Main_Type = "AFM" and NA.Date_Of_Trans >= :BeginDate
      and NA.Date_Of_Trans <= :EndDate
  INTO :ExecutedAFM;

  SELECT Sum(A.Fuel$), Sum(A.Flt_Hrs)
  FROM NAS_CV N, Deployed D, Squadron S, ASKIT A
  WHERE N.NAS_CV_Name = :NASName and N.NAS_CV_UIC = D.NAS_CV_UIC and
D.Squadron_UIC = S.Squadron_UIC and
  S.Squadron_UIC = A.Squadron_UIC and A.Trans_Start_Date >= :BeginDate and A.Trans_End_Date
<= :EndDate
  INTO :ExecutedFuel, :ExecutedFltHrs;

  ShouldAVDLR = 0.0;
  ShouldAFM   = 0.0;
  ShouldFuel  = 0.0;

  FOR
    SELECT S.Squadron_UIC
    FROM NAS_CV N, Deployed D, Squadron S
    WHERE N.NAS_CV_Name = :NASName and N.NAS_CV_UIC = D.NAS_CV_UIC and
D.Squadron_UIC = S.Squadron_UIC
    INTO :SQD
  DO
    BEGIN
      SELECT Should_AVDLR, Should_AFM, Should_Fuel
      FROM SHOULD_COST(:SQD, :FY, :BeginDate, :EndDate)
      INTO :X, :Y, :Z;

      ShouldAVDLR =  X + ShouldAVDLR;
      ShouldAFM   =  Y + ShouldAFM;
      ShouldFuel  =  Z + ShouldFuel;

    END
 SUSPEND;
END^
SET TERM ; ^
```

H. STORED PROCEDURE - NAS_SQUADRON SUMMARY

```
CONNECT "d:\thesis\database\afast_db"
USER"sqldba" PASSWORD "masterkey";
SET TERM ^ ;

CREATE PROCEDURE NAS_Sqd_Summary(SquadronName Char(25), NASName Char(25),FY
Integer, BeginDate Date, EndDate Date, Quarter Integer) RETURNS (AllocatedFuel Float,
AllocatedFltHrs Integer, ExecutedAVDLR Float, ExecutedAFM Float, ExecutedFuel Float,
ExecutedFltHrs Integer, ShouldAVDLR Float, ShouldAFM Float, ShouldFuel Float) AS

DECLARE VARIABLE SQD_UIC CHAR(6);
DECLARE VARIABLE NAS_UIC CHAR(6);
```

```
BEGIN

  SELECT NAS_CV_UIC
  FROM NAS_CV
  WHERE NAS_CV_NAME = :NASName
  INTO :NAS_UIC;

  SELECT Squadron_UIC
  FROM Squadron
  WHERE Squadron_Name = :SquadronName
  INTO :SQD_UIC;

  IF (:Quarter = 0) then
  BEGIN
    SELECT Sum(Fuel_Total), Sum(Hours_Total)
    FROM SQD_Allocated SA, Squadron S
    WHERE S.Squadron_Name = :SquadronName and S.Squadron_UIC = SA.Squadron_UIC and
SA.Fiscal_Year = :FY
    INTO :AllocatedFuel, :AllocatedFltHrs;
  END

  IF (:Quarter = 1) then
  BEGIN
    SELECT Sum(Fuel_Q1), Sum(Hours_Q1)
    FROM SQD_Allocated SA, Squadron S
    WHERE S.Squadron_Name = :SquadronName and S.Squadron_UIC = SA.Squadron_UIC and
SA.Fiscal_Year = :FY
    INTO :AllocatedFuel, :AllocatedFltHrs;
  END

  IF (:Quarter = 2) then
  BEGIN
    SELECT Sum(Fuel_Q2), Sum(Hours_Q2)
    FROM SQD_Allocated SA, Squadron S
    WHERE S.Squadron_Name = :SquadronName and S.Squadron_UIC = SA.Squadron_UIC and
SA.Fiscal_Year = :FY
    INTO :AllocatedFuel, :AllocatedFltHrs;
  END

  IF (:Quarter = 3) then
  BEGIN
    SELECT Sum(Fuel_Q3), Sum(Hours_Q3)
    FROM SQD_Allocated SA, Squadron S
    WHERE S.Squadron_Name = :SquadronName and S.Squadron_UIC = SA.Squadron_UIC and
SA.Fiscal_Year = :FY
    INTO :AllocatedFuel, :AllocatedFltHrs;
  END

  IF (:Quarter = 4) then
  BEGIN
    SELECT Sum(Fuel_Q4), Sum(Hours_Q4)
    FROM SQD_Allocated SA, Squadron S
    WHERE S.Squadron_Name = :SquadronName and S.Squadron_UIC = SA.Squadron_UIC and
SA.Fiscal_Year = :FY
```

```
    INTO :AllocatedFuel, :AllocatedFltHrs;
  END

  SELECT Sum(Cost)
  FROM Squadron S, NACA NA
  WHERE S.Squadron_Name = :SquadronName and S.Squadron_UIC = NA.Squadron_UIC and
NA.NAS_CV_UIC = :NAS_UIC and NA.Main_Type = "AVDLR" and NA.Date_Of_Trans >=
:BeginDate
      and Date_Of_Trans <= :EndDate
  INTO :ExecutedAVDLR;

  SELECT Sum(Cost)
  FROM Squadron S, NACA NA
  WHERE S.Squadron_Name = :SquadronName and S.Squadron_UIC = NA.Squadron_UIC and
NA.NAS_CV_UIC = :NAS_UIC and NA.Main_Type = "AFM" and NA.Date_Of_Trans >= :BeginDate
      and Date_Of_Trans <= :EndDate
  INTO :ExecutedAFM;

  SELECT Sum(Fuel$)
  FROM Squadron S, ASKIT A
  WHERE S.Squadron_Name = :SquadronName and S.Squadron_UIC = A.Squadron_UIC and
A.NAS_CV_UIC = :NAS_UIC and A.Trans_Start_Date >= :BeginDate and A.Trans_End_Date <=
:EndDate
  INTO :ExecutedFuel;

  SELECT Sum(Flt_Hrs)
  FROM Squadron S, ASKIT A
  WHERE S.Squadron_Name = :SquadronName and S.Squadron_UIC = A.Squadron_UIC and
A.NAS_CV_UIC = :NAS_UIC and A.Trans_Start_Date >= :BeginDate and A.Trans_End_Date <=
:EndDate
  INTO :ExecutedFltHrs;

  SELECT Should_AVDLR, Should_AFM, Should_Fuel
  FROM SHOULD_COST(:SQD_UIC, :FY, :BeginDate, :EndDate)
  INTO :ShouldAVDLR, :ShouldAFM, :ShouldFuel;

 SUSPEND;
END^
SET TERM ; ^
```

I. STORED PROCEDURE - NAS_SQUADRON DETAIL

```
CONNECT "d:\thesis\database\afast_db"
USER"sqldba" PASSWORD "masterkey";
SET TERM ^ ;

CREATE PROCEDURE NAS_DETAIL(SquadronName Char(25), NASCVName Char(25),TMS_NO
Integer,FY Integer, BeginDate Date, EndDate Date) RETURNS
(ExecutedAVDLR Float, AVDLRPwrPlants Float, AVDLRAvionics Float, AVDLRAirFrames Float,
AVDLROther Float, AVDLRMiscTEC Float, AVDLROverhead Float,ExecutedAFM Float,
AFMPwrPlants Float, AFMAvionics Float, AFMAirFrames Float, AFMOther Float, AFMMiscTEC Float,
```

```
AFMOverhead Float, ExecutedFuel Float, ExecutedFltHrs Integer, ShouldAVDLR Float, ShouldAFM
Float, ShouldFuel Float, CostHrAVDLR Integer, CostHrAFM Integer, CostHrFuel Integer,
ActualCostHrAVDLR Float, ActualCostHrAFM Float,ActualCostHrFuel Float) AS

DECLARE VARIABLE NASCVUIC Char(6);
BEGIN

  SELECT NAS_CV_UIC
  FROM NAS_CV
  WHERE NAS_CV_NAME = :NASCVName
  INTO :NASCVUIC;

  SELECT Sum(Cost)
  FROM Squadron S, NACA NA
  WHERE S.Squadron_Name = :SquadronName and S.Squadron_UIC = NA.Squadron_UIC and
NA.ID_TMS = :TMS_NO and NA.NAS_CV_UIC =:NASCVUIC and NA.Main_Type = "AVDLR"  and
NA.Date_Of_Trans >= :BeginDate
      and NA.Date_Of_Trans <= :EndDate
  INTO :ExecutedAVDLR;

  SELECT Sum(Cost)
  FROM Squadron S, NACA NA
  WHERE S.Squadron_Name = :SquadronName and S.Squadron_UIC = NA.Squadron_UIC and
NA.ID_TMS = :TMS_NO and NA.NAS_CV_UIC =:NASCVUIC and NA.Main_Type = "AVDLR" and
NA.BRANCH = "PWR_PLANTS" and NA.Date_Of_Trans >= :BeginDate
      and NA.Date_Of_Trans <= :EndDate
  INTO :AVDLRPwrPlants;

  SELECT Sum(Cost)
  FROM Squadron S, NACA NA
  WHERE S.Squadron_Name = :SquadronName and S.Squadron_UIC = NA.Squadron_UIC and
NA.ID_TMS = :TMS_NO and NA.NAS_CV_UIC =:NASCVUIC and NA.Main_Type = "AVDLR" and
NA.BRANCH = "AVIONICS" and NA.Date_Of_Trans >= :BeginDate
      and NA.Date_Of_Trans <= :EndDate
  INTO :AVDLRAvionics;

  SELECT Sum(Cost)
  FROM Squadron S, NACA NA
  WHERE S.Squadron_Name = :SquadronName and S.Squadron_UIC = NA.Squadron_UIC and
NA.ID_TMS = :TMS_NO and NA.NAS_CV_UIC =:NASCVUIC and NA.Main_Type = "AVDLR" and
NA.BRANCH = "AIRFRAMES" and NA.Date_Of_Trans >= :BeginDate
      and NA.Date_Of_Trans <= :EndDate
  INTO :AVDLRAirFrames;

  SELECT Sum(Cost)
  FROM Squadron S, NACA NA
  WHERE S.Squadron_Name = :SquadronName and S.Squadron_UIC = NA.Squadron_UIC and
NA.ID_TMS = :TMS_NO and NA.NAS_CV_UIC =:NASCVUIC and NA.Main_Type = "AVDLR" and
NA.BRANCH = "OTHER" and NA.Date_Of_Trans >= :BeginDate
      and NA.Date_Of_Trans <= :EndDate
  INTO :AVDLROther;

  SELECT Sum(Cost)
  FROM Squadron S, NACA NA
```

```
  WHERE S.Squadron_Name = :SquadronName and S.Squadron_UIC = NA.Squadron_UIC and
NA.ID_TMS = :TMS_NO and NA.NAS_CV_UIC =:NASCVUIC and NA.Main_Type = "AVDLR" and
NA.BRANCH = "MISC_TEC" and NA.Date_Of_Trans >= :BeginDate
      and NA.Date_Of_Trans <= :EndDate
  INTO :AVDLRMiscTEC;

  SELECT Sum(Cost)
  FROM Squadron S, NACA NA
  WHERE S.Squadron_Name = :SquadronName and S.Squadron_UIC = NA.Squadron_UIC and
NA.ID_TMS = :TMS_NO and NA.NAS_CV_UIC =:NASCVUIC and NA.Main_Type = "AVDLR" and
NA.BRANCH = "OVERHEAD" and NA.Date_Of_Trans >= :BeginDate
      and NA.Date_Of_Trans <= :EndDate
  INTO :AVDLROverhead;

  SELECT Sum(Cost)
  FROM Squadron S, NACA NA
  WHERE S.Squadron_Name = :SquadronName and S.Squadron_UIC = NA.Squadron_UIC and
NA.ID_TMS = :TMS_NO and NA.NAS_CV_UIC =:NASCVUIC and NA.Main_Type = "AFM"  and
NA.Date_Of_Trans >= :BeginDate
      and NA.Date_Of_Trans <= :EndDate
  INTO :ExecutedAFM;

  SELECT Sum(Cost)
  FROM Squadron S, NACA NA
  WHERE S.Squadron_Name = :SquadronName and S.Squadron_UIC = NA.Squadron_UIC and
NA.ID_TMS = :TMS_NO and NA.NAS_CV_UIC =:NASCVUIC and NA.Main_Type = "AFM" and
NA.BRANCH = "PWR_PLANTS" and NA.Date_Of_Trans >= :BeginDate
      and NA.Date_Of_Trans <= :EndDate
  INTO :AFMPwrPlants;

  SELECT Sum(Cost)
  FROM Squadron S, NACA NA
  WHERE S.Squadron_Name = :SquadronName and S.Squadron_UIC = NA.Squadron_UIC and
NA.ID_TMS = :TMS_NO and NA.NAS_CV_UIC =:NASCVUIC and NA.Main_Type = "AFM" and
NA.BRANCH = "AVIONICS" and NA.Date_Of_Trans >= :BeginDate
      and NA.Date_Of_Trans <= :EndDate
  INTO :AFMAvionics;

  SELECT Sum(Cost)
  FROM Squadron S, NACA NA
  WHERE S.Squadron_Name = :SquadronName and S.Squadron_UIC = NA.Squadron_UIC and
NA.ID_TMS = :TMS_NO and NA.NAS_CV_UIC =:NASCVUIC and NA.Main_Type = "AFM" and
NA.BRANCH = "AIRFRAMES" and NA.Date_Of_Trans >= :BeginDate
      and NA.Date_Of_Trans <= :EndDate
  INTO :AFMAirFrames;

  SELECT Sum(Cost)
  FROM Squadron S, NACA NA
  WHERE S.Squadron_Name = :SquadronName and S.Squadron_UIC = NA.Squadron_UIC and
NA.ID_TMS = :TMS_NO and NA.NAS_CV_UIC =:NASCVUIC and NA.Main_Type = "AFM" and
NA.BRANCH = "OTHER" and NA.Date_Of_Trans >= :BeginDate
      and NA.Date_Of_Trans <= :EndDate
  INTO :AFMOther;
```

```
  SELECT Sum(Cost)
  FROM Squadron S, NACA NA
  WHERE S.Squadron_Name = :SquadronName and S.Squadron_UIC = NA.Squadron_UIC and
NA.ID_TMS = :TMS_NO and NA.NAS_CV_UIC =:NASCVUIC and NA.Main_Type = "AFM" and
NA.BRANCH = "MISC_TEC" and NA.Date_Of_Trans >= :BeginDate
      and NA.Date_Of_Trans <= :EndDate
  INTO :AFMMiscTEC;

  SELECT Sum(Cost)
  FROM Squadron S, NACA NA
  WHERE S.Squadron_Name = :SquadronName and S.Squadron_UIC = NA.Squadron_UIC and
NA.ID_TMS = :TMS_NO and NA.NAS_CV_UIC =:NASCVUIC and NA.Main_Type = "AFM" and
NA.BRANCH = "OVERHEAD" and NA.Date_Of_Trans >= :BeginDate
      and NA.Date_Of_Trans <= :EndDate
  INTO :AFMOverhead;

  SELECT Sum(Fuel$), Sum(A.Flt_Hrs)
  FROM Squadron S, ASKIT A
  WHERE S.Squadron_Name = :SquadronName and S.Squadron_UIC = A.Squadron_UIC and
A.ID_TMS = :TMS_NO and A.NAS_CV_UIC =:NASCVUIC and A.Trans_Start_Date >= :BeginDate
and A.Trans_End_Date <= :EndDate
  INTO :ExecutedFuel, :ExecutedFltHrs;

  SELECT T.AVDLR_Cost_Hour, T.AFM_Cost_Hour, T.Fuel_Cost_Hour
  FROM TMS T
  WHERE  T.ID_TMS = :TMS_NO
  INTO :CostHrAVDLR, :CostHrAFM, :CostHrFuel;

  if (ExecutedFltHrs is NULL) then
    ExecutedFltHrs = 0;

  ShouldAVDLR = ExecutedFltHrs * CostHrAVDLR;
  ShouldAFM = ExecutedFltHrs * CostHrAFM;
  ShouldFuel = ExecutedFltHrs * CostHrFuel;

  ActualCostHrAVDLR = ExecutedAVDLR / ExecutedFltHrs;
  ActualCostHrAFM   = ExecutedAFM / ExecutedFltHrs;
  ActualCostHrFuel  = ExecutedFuel / ExecutedFltHrs;

SUSPEND;
END ^
SET TERM ; ^
```

J. STORED PROCEDURE - ORGANIZATION SUMMARY

```
CONNECT "d:\thesis\database\afast_db"
USER"sqldba" PASSWORD "masterkey";
SET TERM ^ ;

CREATE PROCEDURE ORG_Summary(ORGName Char(25), NASName Char(25), FY Integer,
BeginDate Date, EndDate Date) RETURNS (AVDLRPwrPlants Float, AVDLRAvionics Float,
AVDLRAirFrames Float, AVDLROther Float,AVDLRMiscTEC Float, AVDLROverhead Float,
```

```
AFMPwrPlants Float, AFMAvionics Float, AFMAirFrames Float, AFMOther Float, AFMMiscTEC Float,
AFMOverhead Float) AS

DECLARE VARIABLE NASUIC Char(6);
BEGIN

  SELECT NAS_CV_UIC
  FROM NAS_CV
  WHERE NAS_CV_NAME = :NASName
  INTO :NASUIC;

  SELECT Sum(Cost)
  FROM NACA NA
  WHERE NA.NAS_CV_UIC = :NASUIC and NA.ORG_Name = :ORGName and NA.Main_Type =
"AVDLR" and NA.BRANCH = "PWR_PLANTS" and NA.Date_Of_Trans >= :BeginDate
      and NA.Date_Of_Trans <= :EndDate
  INTO :AVDLRPwrPlants;

  SELECT Sum(Cost)
  FROM NACA NA
  WHERE NA.NAS_CV_UIC = :NASUIC and NA.ORG_Name = :ORGName and NA.Main_Type =
"AVDLR" and NA.BRANCH = "AVIONICS" and NA.Date_Of_Trans >= :BeginDate
      and NA.Date_Of_Trans <= :EndDate
  INTO :AVDLRAvionics;

  SELECT Sum(Cost)
  FROM NACA NA
  WHERE NA.NAS_CV_UIC = :NASUIC and NA.ORG_Name = :ORGName and NA.Main_Type =
"AVDLR" and NA.BRANCH = "AIRFRAMES" and NA.Date_Of_Trans >= :BeginDate
      and NA.Date_Of_Trans <= :EndDate
  INTO :AVDLRAirFrames;

  SELECT Sum(Cost)
  FROM NACA NA
  WHERE NA.NAS_CV_UIC = :NASUIC and NA.ORG_Name = :ORGName and NA.Main_Type =
"AVDLR" and NA.BRANCH = "OTHER" and NA.Date_Of_Trans >= :BeginDate
      and NA.Date_Of_Trans <= :EndDate
  INTO :AVDLROther;

  SELECT Sum(Cost)
  FROM NACA NA
  WHERE NA.NAS_CV_UIC = :NASUIC and NA.ORG_Name = :ORGName and NA.Main_Type =
"AVDLR" and NA.BRANCH = "MISC_TEC" and NA.Date_Of_Trans >= :BeginDate
      and NA.Date_Of_Trans <= :EndDate
  INTO :AVDLRMiscTEC;

  SELECT Sum(Cost)
  FROM NACA NA
  WHERE NA.NAS_CV_UIC = :NASUIC and NA.ORG_Name = :ORGName and NA.Main_Type =
"AVDLR" and NA.BRANCH = "OVERHEAD" and NA.Date_Of_Trans >= :BeginDate
      and NA.Date_Of_Trans <= :EndDate
  INTO :AVDLROverhead;

  SELECT Sum(Cost)
```

```
  FROM NACA NA
  WHERE NA.NAS_CV_UIC = :NASUIC and NA.ORG_Name = :ORGName and NA.Main_Type =
"AFM" and NA.BRANCH = "PWR_PLANTS" and NA.Date_Of_Trans >= :BeginDate
      and NA.Date_Of_Trans <= :EndDate
  INTO :AFMPwrPlants;

  SELECT Sum(Cost)
  FROM NACA NA
  WHERE NA.NAS_CV_UIC = :NASUIC and NA.ORG_Name = :ORGName and NA.Main_Type =
"AFM" and NA.BRANCH = "AVIONICS" and NA.Date_Of_Trans >= :BeginDate
      and NA.Date_Of_Trans <= :EndDate
  INTO :AFMAvionics;

  SELECT Sum(Cost)
  FROM NACA NA
  WHERE NA.NAS_CV_UIC = :NASUIC and NA.ORG_Name = :ORGName and NA.Main_Type =
"AFM" and NA.BRANCH = "AIRFRAMES" and NA.Date_Of_Trans >= :BeginDate
      and NA.Date_Of_Trans <= :EndDate
  INTO :AFMAirFrames;

  SELECT Sum(Cost)
  FROM NACA NA
  WHERE NA.NAS_CV_UIC = :NASUIC and NA.ORG_Name = :ORGName and NA.Main_Type =
"AFM" and NA.BRANCH = "OTHER" and NA.Date_Of_Trans >= :BeginDate
      and NA.Date_Of_Trans <= :EndDate
  INTO :AFMOther;

  SELECT Sum(Cost)
  FROM NACA NA
  WHERE NA.NAS_CV_UIC = :NASUIC and NA.ORG_Name = :ORGName and NA.Main_Type =
"AFM" and NA.BRANCH = "MISC_TEC" and NA.Date_Of_Trans >= :BeginDate
      and NA.Date_Of_Trans <= :EndDate
  INTO :AFMMiscTEC;

  SELECT Sum(Cost)
  FROM NACA NA
  WHERE NA.NAS_CV_UIC = :NASUIC and NA.ORG_Name = :ORGName and NA.Main_Type =
"AFM" and NA.BRANCH = "OVERHEAD" and NA.Date_Of_Trans >= :BeginDate
      and NA.Date_Of_Trans <= :EndDate
  INTO :AFMOverhead;

SUSPEND;
END ^
SET TERM ; ^
```

K. STORED PROCEDURE - SHOULD COST

```
/* This procedure calculates the should-cost of  a squadron that is past in as a parameter */

CONNECT "d:\thesis\database\afast_db"
USER"sqldba" PASSWORD "masterkey";
SET TERM ^ ;
```

```
CREATE PROCEDURE SHOULD_COST(SQD CHAR(6), FY Integer, BeginDate Date, EndDate Date)
RETURNS (Should_AVDLR Float, Should_AFM Float, Should_Fuel Float) AS

DECLARE VARIABLE TMS_NO      Integer;
DECLARE VARIABLE CostHrAVDLR Integer;
DECLARE VARIABLE CostHrAFM   Integer;
DECLARE VARIABLE CostHrFuel  Integer;
DECLARE VARIABLE SumFltHrs   Integer;
BEGIN
  Should_AVDLR = 0.0;
  Should_AFM   = 0.0;
  Should_Fuel  = 0.0;

      /* For each squadron TMS  obtain the cost per flight hour and calculate the should cost */
      FOR
        SELECT T.ID_TMS, T.AVDLR_Cost_Hour, T.AFM_Cost_Hour, T.Fuel_Cost_Hour
        FROM Squadron S, SQD_Allocated SA, TMS T
        WHERE  S.Squadron_UIC = :SQD and S.Squadron_UIC = SA.Squadron_UIC and SA.ID_TMS =
            T.ID_TMS and T.Fiscal_Year = :FY
        INTO :TMS_NO, :CostHrAVDLR, :CostHrAFM, :CostHrFuel
      DO
        BEGIN
        if (:TMS_NO is NULL) then
          BEGIN
            CostHrAVDLR = 0;
            CostHrAFM = 0;
            CostHrFuel = 0;
          END

          SELECT Sum(A.Flt_Hrs)
          FROM Squadron S, ASKIT A
          WHERE S.Squadron_UIC = :SQD and S.Squadron_UIC = A.Squadron_UIC and A.ID_TMS =
              :TMS_NO and Trans_Start_Date >= :BeginDate and Trans_End_Date <= :EndDate
          INTO :SumFltHrs;

          if (:SumFltHrs is NULL) then
            SumFltHrs = 0;

          Should_AVDLR = CostHrAVDLR * SumFltHrs + Should_AVDLR;
          Should_AFM   = CostHrAFM   * SumFltHrs + Should_AFM;
          Should_Fuel  = CostHrFuel  * SumFltHrs + Should_Fuel;
        END
SUSPEND;
END^
set term : ^
```

L. STORED PROCEDURE - SQUADRON SUMMARY

```
CONNECT "d:\thesis\database\afast_db"
USER"sqldba" PASSWORD "masterkey";
SET TERM ^ ;
```

```
CREATE PROCEDURE Squadron_Summary(SquadronName Char(25), FY Integer, BeginDate Date,
EndDate Date, Quarter Integer) RETURNS
  (AllocatedFuel Float, AllocatedFltHrs Integer,
   ExecutedAVDLR Float, ExecutedAFM Float, ExecutedFuel Float, ExecutedFltHrs Integer,
   ShouldAVDLR Float, ShouldAFM Float, ShouldFuel Float) AS

DECLARE VARIABLE SQD_UIC CHAR(6);

BEGIN
  if (:Quarter = 0) then
  BEGIN
    SELECT Sum(Fuel_Total), Sum(Hours_Total)
    FROM SQD_Allocated SA, Squadron S
    WHERE S.Squadron_Name = :SquadronName and S.Squadron_UIC = SA.Squadron_UIC and
SA.Fiscal_Year = :FY
    INTO :AllocatedFuel, :AllocatedFltHrs;
  END

  if (:Quarter = 1) then
  BEGIN
    SELECT Sum(Fuel_Q1), Sum(Hours_Q1)
    FROM SQD_Allocated SA, Squadron S
    WHERE S.Squadron_Name = :SquadronName and S.Squadron_UIC = SA.Squadron_UIC and
SA.Fiscal_Year = :FY
    INTO :AllocatedFuel, :AllocatedFltHrs;
  END

  if (:Quarter = 2) then
  BEGIN
    SELECT Sum(Fuel_Q2), Sum(Hours_Q2)
    FROM SQD_Allocated SA, Squadron S
    WHERE S.Squadron_Name = :SquadronName and S.Squadron_UIC = SA.Squadron_UIC and
SA.Fiscal_Year = :FY
    INTO :AllocatedFuel, :AllocatedFltHrs;
  END

  if (:Quarter = 3) then
  BEGIN
    SELECT Sum(Fuel_Q3), Sum(Hours_Q3)
    FROM SQD_Allocated SA, Squadron S
    WHERE S.Squadron_Name = :SquadronName and S.Squadron_UIC = SA.Squadron_UIC and
SA.Fiscal_Year = :FY
    INTO :AllocatedFuel, :AllocatedFltHrs;
  END

  if (:Quarter = 4) then
  BEGIN
    SELECT Sum(Fuel_Q4), Sum(Hours_Q4)
    FROM SQD_Allocated SA, Squadron S
    WHERE S.Squadron_Name = :SquadronName and S.Squadron_UIC = SA.Squadron_UIC and
SA.Fiscal_Year = :FY
    INTO :AllocatedFuel, :AllocatedFltHrs;
  END
```

```
  SELECT Sum(Cost)
  FROM Squadron S, NACA NA
  WHERE S.Squadron_Name = :SquadronName and S.Squadron_UIC = NA.Squadron_UIC and
NA.Main_Type = "AVDLR" and NA.Date_Of_Trans >= :BeginDate
      and Date_Of_Trans <= :EndDate
  INTO :ExecutedAVDLR;

  SELECT Sum(Cost)
  FROM Squadron S, NACA NA
  WHERE S.Squadron_Name = :SquadronName and S.Squadron_UIC = NA.Squadron_UIC and
NA.Main_Type = "AFM" and NA.Date_Of_Trans >= :BeginDate
      and Date_Of_Trans <= :EndDate
  INTO :ExecutedAFM;

  SELECT Sum(Fuel$)
  FROM Squadron S, ASKIT A
  WHERE S.Squadron_Name = :SquadronName and S.Squadron_UIC = A.Squadron_UIC and
A.Trans_Start_Date >= :BeginDate and A.Trans_End_Date <= :EndDate
  INTO :ExecutedFuel;

  SELECT Sum(Flt_Hrs)
  FROM Squadron S, ASKIT A
  WHERE S.Squadron_Name = :SquadronName and S.Squadron_UIC = A.Squadron_UIC and
A.Trans_Start_Date >= :BeginDate and A.Trans_End_Date <= :EndDate
  INTO :ExecutedFltHrs;

  SELECT Squadron_UIC
  FROM Squadron
  WHERE Squadron_Name = :SquadronName
  INTO :SQD_UIC;

  SELECT Should_AVDLR, Should_AFM, Should_Fuel
  FROM SHOULD_COST(:SQD_UIC, :FY, :BeginDate, :EndDate)
  INTO :ShouldAVDLR, :ShouldAFM, :ShouldFuel;

SUSPEND;
END^
SET TERM ; ^
```

M. STORED PROCEDURE - SQUADRON DETAIL

```
CONNECT "d:\thesis\database\afast_db"
USER"sqldba" PASSWORD "masterkey";
SET TERM ^ ;

CREATE PROCEDURE SQD_DETAIL(SquadronName Char(25), TMS_NO Integer,FY Integer,
BeginDate Date, EndDate Date) RETURNS
(ExecutedAVDLR Float, AVDLRPwrPlants Float, AVDLRAvionics Float, AVDLRAirFrames Float,
AVDLROther Float, AVDLRMiscTEC Float, AVDLROverhead Float,
ExecutedAFM Float, AFMPwrPlants Float, AFMAvionics Float, AFMAirFrames Float, AFMOther Float,
AFMMiscTEC Float, AFMOverhead Float, ExecutedFuel Float, ExecutedFltHrs Integer, ShouldAVDLR
```

```
Float, ShouldAFM Float, ShouldFuel Float, CostHrAVDLR Integer, CostHrAFM Integer, CostHrFuel
Integer, ActualCostHrAVDLR Float, ActualCostHrAFM Float,ActualCostHrFuel Float) AS

BEGIN

  SELECT Sum(Cost)
  FROM Squadron S, NACA NA
  WHERE S.Squadron_Name = :SquadronName and S.Squadron_UIC = NA.Squadron_UIC and
NA.ID_TMS = :TMS_NO and NA.Main_Type = "AVDLR"  and NA.Date_Of_Trans >= :BeginDate
      and NA.Date_Of_Trans <= :EndDate
  INTO :ExecutedAVDLR;

  SELECT Sum(Cost)
  FROM Squadron S, NACA NA
  WHERE S.Squadron_Name = :SquadronName and S.Squadron_UIC = NA.Squadron_UIC and
NA.ID_TMS = :TMS_NO and NA.Main_Type = "AVDLR" and NA.BRANCH = "PWR_PLANTS" and
NA.Date_Of_Trans >= :BeginDate
      and NA.Date_Of_Trans <= :EndDate
  INTO :AVDLRPwrPlants;

  SELECT Sum(Cost)
  FROM Squadron S, NACA NA
  WHERE S.Squadron_Name = :SquadronName and S.Squadron_UIC = NA.Squadron_UIC and
NA.ID_TMS = :TMS_NO and NA.Main_Type = "AVDLR" and NA.BRANCH = "AVIONICS" and
NA.Date_Of_Trans >= :BeginDate
      and NA.Date_Of_Trans <= :EndDate
  INTO :AVDLRAvionics;

  SELECT Sum(Cost)
  FROM Squadron S, NACA NA
  WHERE S.Squadron_Name = :SquadronName and S.Squadron_UIC = NA.Squadron_UIC and
NA.ID_TMS = :TMS_NO and NA.Main_Type = "AVDLR" and NA.BRANCH = "AIRFRAMES" and
NA.Date_Of_Trans >= :BeginDate
      and NA.Date_Of_Trans <= :EndDate
  INTO :AVDLRAirFrames;

  SELECT Sum(Cost)
  FROM Squadron S, NACA NA
  WHERE S.Squadron_Name = :SquadronName and S.Squadron_UIC = NA.Squadron_UIC and
NA.ID_TMS = :TMS_NO and NA.Main_Type = "AVDLR" and NA.BRANCH = "OTHER" and
NA.Date_Of_Trans >= :BeginDate
      and NA.Date_Of_Trans <= :EndDate
  INTO :AVDLROther;

  SELECT Sum(Cost)
  FROM Squadron S, NACA NA
  WHERE S.Squadron_Name = :SquadronName and S.Squadron_UIC = NA.Squadron_UIC and
NA.ID_TMS = :TMS_NO and NA.Main_Type = "AVDLR" and NA.BRANCH = "MISC_TEC" and
NA.Date_Of_Trans >= :BeginDate
      and NA.Date_Of_Trans <= :EndDate
  INTO :AVDLRMiscTEC;

if (AVDLRMiscTEC is NULL) then
    AVDLRMiscTEC = 0.0;
```

```
  SELECT Sum(Cost)
  FROM Squadron S, NACA NA
  WHERE S.Squadron_Name = :SquadronName and S.Squadron_UIC = NA.Squadron_UIC and
NA.ID_TMS = :TMS_NO and NA.Main_Type = "AVDLR" and NA.BRANCH = "OVERHEAD" and
NA.Date_Of_Trans >= :BeginDate
      and NA.Date_Of_Trans <= :EndDate
  INTO :AVDLROverhead;

if (AVDLROverhead is NULL) then
    AVDLROverhead = 0.0;

  SELECT Sum(Cost)
  FROM Squadron S, NACA NA
  WHERE S.Squadron_Name = :SquadronName and S.Squadron_UIC = NA.Squadron_UIC and
NA.ID_TMS = :TMS_NO and NA.Main_Type = "AFM"  and NA.Date_Of_Trans >= :BeginDate
      and NA.Date_Of_Trans <= :EndDate
  INTO :ExecutedAFM;

  SELECT Sum(Cost)
  FROM Squadron S, NACA NA
  WHERE S.Squadron_Name = :SquadronName and S.Squadron_UIC = NA.Squadron_UIC and
NA.ID_TMS = :TMS_NO and NA.Main_Type = "AFM" and NA.BRANCH = "PWR_PLANTS" and
NA.Date_Of_Trans >= :BeginDate
      and NA.Date_Of_Trans <= :EndDate
  INTO :AFMPwrPlants;

  SELECT Sum(Cost)
  FROM Squadron S, NACA NA
  WHERE S.Squadron_Name = :SquadronName and S.Squadron_UIC = NA.Squadron_UIC and
NA.ID_TMS = :TMS_NO and NA.Main_Type = "AFM" and NA.BRANCH = "AVIONICS" and
NA.Date_Of_Trans >= :BeginDate
      and NA.Date_Of_Trans <= :EndDate
  INTO :AFMAvionics;

  SELECT Sum(Cost)
  FROM Squadron S, NACA NA
  WHERE S.Squadron_Name = :SquadronName and S.Squadron_UIC = NA.Squadron_UIC and
NA.ID_TMS = :TMS_NO and NA.Main_Type = "AFM" and NA.BRANCH = "AIRFRAMES" and
NA.Date_Of_Trans >= :BeginDate
      and NA.Date_Of_Trans <= :EndDate
  INTO :AFMAirFrames;

  SELECT Sum(Cost)
  FROM Squadron S, NACA NA
  WHERE S.Squadron_Name = :SquadronName and S.Squadron_UIC = NA.Squadron_UIC and
NA.ID_TMS = :TMS_NO and NA.Main_Type = "AFM" and NA.BRANCH = "OTHER" and
NA.Date_Of_Trans >= :BeginDate
      and NA.Date_Of_Trans <= :EndDate
  INTO :AFMOther;


 SELECT Sum(Cost)
  FROM Squadron S, NACA NA
```

```
  WHERE S.Squadron_Name = :SquadronName and S.Squadron_UIC = NA.Squadron_UIC and
NA.ID_TMS = :TMS_NO and NA.Main_Type = "AFM" and NA.BRANCH = "MISC_TEC" and
NA.Date_Of_Trans >= :BeginDate
      and NA.Date_Of_Trans <= :EndDate
  INTO :AFMMiscTEC;

if (AFMMiscTEC is NULL) then
    AFMMiscTEC = 0.0;

  SELECT Sum(Cost)
  FROM Squadron S, NACA NA
  WHERE S.Squadron_Name = :SquadronName and S.Squadron_UIC = NA.Squadron_UIC and
NA.ID_TMS = :TMS_NO and NA.Main_Type = "AFM" and NA.BRANCH = "OVERHEAD" and
NA.Date_Of_Trans >= :BeginDate
      and NA.Date_Of_Trans <= :EndDate
  INTO :AFMOverhead;

if (AFMOverhead is NULL) then
    AFMOverhead = 0.0;

  SELECT Sum(Fuel$), Sum(A.Flt_Hrs)
  FROM Squadron S, ASKIT A
  WHERE S.Squadron_Name = :SquadronName and S.Squadron_UIC = A.Squadron_UIC and
A.ID_TMS = :TMS_NO and A.Trans_Start_Date >= :BeginDate and A.Trans_End_Date <= :EndDate
  INTO :ExecutedFuel, :ExecutedFltHrs;

  SELECT T.AVDLR_Cost_Hour, T.AFM_Cost_Hour, T.Fuel_Cost_Hour
  FROM TMS T
  WHERE  T.ID_TMS = :TMS_NO
  INTO :CostHrAVDLR, :CostHrAFM, :CostHrFuel;

  if (ExecutedFltHrs is NULL) then
    ExecutedFltHrs = 0;

  ShouldAVDLR = ExecutedFltHrs * CostHrAVDLR;
  ShouldAFM = ExecutedFltHrs * CostHrAFM;
  ShouldFuel = ExecutedFltHrs * CostHrFuel;

  ActualCostHrAVDLR = ExecutedAVDLR / ExecutedFltHrs;
  ActualCostHrAFM   = ExecutedAFM / ExecutedFltHrs;
  ActualCostHrFuel  = ExecutedFuel / ExecutedFltHrs;
SUSPEND;
END ^
SET TERM ; ^
```

N. STORED PROCEDURE - TMS SUMMARY

```
CONNECT "d:\thesis\database\afast_db"
USER"sqldba" PASSWORD "masterkey";
SET TERM ^ ;
```

```
CREATE PROCEDURE TMS_Summary(TMSName Char(7), FY Integer, BeginDate Date, EndDate
Date)
 RETURNS (FRSAVDLR Float, FRSAFM Float, FRSFuel Float, FRSFltHrs Integer,
       SupportAVDLR Float, SupportAFM Float, SupportFuel Float, SupportFltHrs Integer,
       TacairAVDLR Float, TacairAFM Float, TacairFuel Float, TacairFltHrs Integer,
        FRSShouldAVDLR Float, FRSShouldAFM Float, FRSShouldFuel Float,
        SupportShouldAVDLR Float, SupportShouldAFM Float, SupportShouldFuel Float,
        TacairShouldAVDLR Float, TacairShouldAFM Float, TacairShouldFuel Float) AS

DECLARE VARIABLE FRS_ID_TMS Integer;
DECLARE VARIABLE FRSCostHrAVDLR Integer;
DECLARE VARIABLE FRSCostHrAFM   Integer;
DECLARE VARIABLE FRSCostHrFuel  Integer;

DECLARE VARIABLE Support_ID_TMS Integer;
DECLARE VARIABLE SupportCostHrAVDLR Integer;
DECLARE VARIABLE SupportCostHrAFM   Integer;
DECLARE VARIABLE SupportCostHrFuel  Integer;

DECLARE VARIABLE Tacair_ID_TMS Integer;
DECLARE VARIABLE TacairCostHrAVDLR Integer;
DECLARE VARIABLE TacairCostHrAFM   Integer;
DECLARE VARIABLE TacairCostHrFuel  Integer;

DECLARE VARIABLE A Float;
DECLARE VARIABLE B Float;
DECLARE VARIABLE C Float;
DECLARE VARIABLE D Integer;
DECLARE VARIABLE E Integer;
DECLARE VARIABLE F Integer;
DECLARE VARIABLE G Integer;
DECLARE VARIABLE H Float;
DECLARE VARIABLE I Float;
DECLARE VARIABLE J Float;
DECLARE VARIABLE K Integer;
DECLARE VARIABLE L Integer;
DECLARE VARIABLE M Integer;
DECLARE VARIABLE N Integer;

BEGIN

  SELECT T.ID_TMS, T.AVDLR_Cost_Hour, T.AFM_Cost_Hour, T.Fuel_Cost_Hour
  FROM TMS T
  WHERE TMS = :TMSName and Mission = "R" and Fiscal_Year = :FY
  INTO :FRS_ID_TMS, :FRSCostHrAVDLR, :FRSCostHrAFM, :FRSCostHrFuel;

  SELECT Sum(Cost)
  FROM NACA N
  WHERE  N.ID_TMS = :FRS_ID_TMS and N.Main_Type = "AVDLR"
    and N.Date_Of_Trans >= :BeginDate and N.Date_Of_Trans <= :EndDate
  INTO :FRSAVDLR;

  SELECT Sum(Cost)
  FROM NACA N
```

```
  WHERE  N.ID_TMS = :FRS_ID_TMS and N.Main_Type = "AFM"
    and N.Date_Of_Trans >= :BeginDate and N.Date_Of_Trans <= :EndDate
  INTO :FRSAFM;

  SELECT Sum(A.Fuel$), Sum(A.Flt_Hrs)
  FROM  ASKIT A
  WHERE  A.ID_TMS = :FRS_ID_TMS and A.Trans_Start_Date >= :BeginDate and A.Trans_End_Date
<= :EndDate
  INTO :FRSFuel, :FRSFltHrs;

  FRSShouldAVDLR = FRSFltHrs * FRSCostHrAVDLR;
  FRSShouldAFM = FRSFltHrs * FRSCostHrAFM;
  FRSShouldFuel = FRSFltHrs * FRSCostHrFuel;

  SupportAVDLR = 0.0;
  SupportAFM = 0.0;
  SupportFuel = 0.0;
  SupportFltHrs = 0;
  SupportShouldAVDLR = 0.0;
  SupportShouldAFM = 0.0;
  SupportShouldFuel = 0.0;

  FOR
    SELECT T.ID_TMS, T.AVDLR_Cost_Hour, T.AFM_Cost_Hour, T.Fuel_Cost_Hour
    FROM TMS T
    WHERE TMS = :TMSName and Mission = "S" and Fiscal_Year = :FY
    INTO :Support_ID_TMS, :SupportCostHrAVDLR, :SupportCostHrAFM, :SupportCostHrFuel
  DO
  BEGIN
    SELECT Sum(Cost)
    FROM NACA N
    WHERE  N.ID_TMS = :Support_ID_TMS and N.Main_Type = "AVDLR"
      and N.Date_Of_Trans >= :BeginDate and N.Date_Of_Trans <= :EndDate
    INTO :H;

    SELECT Sum(Cost)
    FROM NACA N
    WHERE  N.ID_TMS = :Support_ID_TMS and N.Main_Type = "AFM"
      and N.Date_Of_Trans >= :BeginDate and N.Date_Of_Trans <= :EndDate
    INTO :I;

    SELECT Sum(A.Fuel$), Sum(A.Flt_Hrs)
    FROM  ASKIT A
    WHERE  A.ID_TMS = :Support_ID_TMS and A.Trans_Start_Date >= :BeginDate and
A.Trans_End_Date <= :EndDate
    INTO :J, :K;

    if (:J is NULL) then
      J = 0.0;
    if (:K is NULL) then
      K = 0;

    L = K * SupportCostHrAVDLR;
    M = K * SupportCostHrAFM;
```

```
    N = K * SupportCostHrFuel;

    SupportAVDLR = SupportAVDLR + H;
    SupportAFM = SupportAFM + I;
    SupportFuel = SupportFuel + J;
    SupportFltHrs = SupportFltHrs + K;

    SupportShouldAVDLR = SupportShouldAVDLR + L;
    SupportShouldAFM = SupportShouldAFM + M;
    SupportShouldFuel = SupportShouldFuel + N;
  END
  TacairAVDLR = 0.0;
  TacairAFM = 0.0;
  TacairFuel = 0.0;
  TacairFltHrs = 0;
  TacairShouldAVDLR = 0.0;
  TacairShouldAFM = 0.0;
  TacairShouldFuel = 0.0;

  FOR
    SELECT T.ID_TMS, T.AVDLR_Cost_Hour, T.AFM_Cost_Hour, T.Fuel_Cost_Hour
    FROM TMS T
    WHERE TMS = :TMSName and Mission = "T" and Fiscal_Year = :FY
    INTO :Tacair_ID_TMS, :TacairCostHrAVDLR, :TacairCostHrAFM, :TacairCostHrFuel
  DO
  BEGIN

    SELECT Sum(Cost)
    FROM NACA N
    WHERE  N.ID_TMS = :Tacair_ID_TMS and N.Main_Type = "AVDLR"
      and N.Date_Of_Trans >= :BeginDate and N.Date_Of_Trans <= :EndDate
    INTO :A;

    SELECT Sum(Cost)
    FROM NACA N
    WHERE  N.ID_TMS = :Tacair_ID_TMS and N.Main_Type = "AFM"
      and N.Date_Of_Trans >= :BeginDate and N.Date_Of_Trans <= :EndDate
    INTO :B;

    SELECT Sum(A.Fuel$), Sum(A.Flt_Hrs)
    FROM  ASKIT A
    WHERE  A.ID_TMS = :Tacair_ID_TMS and A.Trans_Start_Date >= :BeginDate and
A.Trans_End_Date <= :EndDate
    INTO :C, :D;

    if (:C is NULL) then
      C = 0.0;
    if (:D is NULL) then
      D = 0;

    E = D * :TacairCostHrAVDLR;
    F = D * :TacairCostHrAFM;
    G = D * :TacairCostHrFuel;
```

```
    TacairAVDLR = TacairAVDLR + A;
    TacairAFM = TacairAFM + B;
    TacairFuel = TacairFuel + C;
    TacairFltHrs = TacairFltHrs + D;

    TacairShouldAVDLR = TacairShouldAVDLR + E;
    TacairShouldAFM = TacairShouldAFM + F;
    TacairShouldFuel = TacairShouldFuel + G;
  END

SUSPEND;
END^
SET TERM ; ^
```

O. STORED PROCEDURE - TMS DETAIL

```
CONNECT "d:\thesis\database\afast_db"
USER"sqldba" PASSWORD "masterkey";
SET TERM ^ ;

CREATE PROCEDURE TMS_DETAIL(TMSName Char(7), FY Integer, BeginDate Date, EndDate
Date)
 RETURNS (FRSAVDLRPwrPlants Float, FRSAVDLRAvionics Float, FRSAVDLRAirFrames Float,
FRSAVDLROther Float, FRSAVDLRMiscTEC Float, FRSAVDLROverhead Float,
FRSAFMPwrPlants Float, FRSAFMAvionics Float, FRSAFMAirFrames Float, FRSAFMOther Float,
FRSAFMMiscTEC Float, FRSAFMOverhead Float,
SupportAVDLRPwrPlants Float, SupportAVDLRAvionics Float, SupportAVDLRAirFrames Float,
SupportAVDLROther Float, SupportAVDLRMiscTEC Float, SupportAVDLROverhead Float,
SupportAFMPwrPlants Float, SupportAFMAvionics Float, SupportAFMAirFrames Float,
SupportAFMOther Float, SupportAFMMiscTEC Float, SupportAFMOverhead Float,
TacairAVDLRPwrPlants Float, TacairAVDLRAvionics Float, TacairAVDLRAirFrames Float,
TacairAVDLROther Float, TacairAVDLRMiscTEC Float, TacairAVDLROverhead Float,
TacairAFMPwrPlants Float, TacairAFMAvionics Float, TacairAFMAirFrames Float, TacairAFMOther
Float, TacairAFMMiscTEC Float, TacairAFMOverhead Float) AS

  DECLARE VARIABLE FRS_TMS_NO Integer;
  DECLARE VARIABLE Support_TMS_NO Integer;
  DECLARE VARIABLE Tacair_TMS_NO Integer;
  DECLARE VARIABLE A FLOAT;
  DECLARE VARIABLE B FLOAT;
  DECLARE VARIABLE C FLOAT;
  DECLARE VARIABLE D FLOAT;
  DECLARE VARIABLE E FLOAT;
  DECLARE VARIABLE F FLOAT;
  DECLARE VARIABLE G FLOAT;
  DECLARE VARIABLE H FLOAT;
  DECLARE VARIABLE I FLOAT;
  DECLARE VARIABLE J FLOAT;
  DECLARE VARIABLE K FLOAT;
  DECLARE VARIABLE L FLOAT;
  DECLARE VARIABLE M FLOAT;
  DECLARE VARIABLE N FLOAT;
```

```
  DECLARE VARIABLE O FLOAT;
  DECLARE VARIABLE P FLOAT;
  DECLARE VARIABLE Q FLOAT;
  DECLARE VARIABLE R FLOAT;
  DECLARE VARIABLE S FLOAT;
  DECLARE VARIABLE T FLOAT;
  DECLARE VARIABLE U FLOAT;
  DECLARE VARIABLE V FLOAT;
  DECLARE VARIABLE W FLOAT;
  DECLARE VARIABLE X FLOAT;

BEGIN

  SELECT ID_TMS
  FROM TMS T
  WHERE T.TMS = :TMSName and T.Mission = "R" and Fiscal_Year = :FY
  INTO :FRS_TMS_NO;

  SELECT Sum(Cost)
  FROM NACA NA
  WHERE NA.ID_TMS = :FRS_TMS_NO and NA.Main_Type = "AVDLR" and NA.BRANCH =
"PWR_PLANTS" and NA.Date_Of_Trans >= :BeginDate
      and NA.Date_Of_Trans <= :EndDate
  INTO :FRSAVDLRPwrPlants;

  SELECT Sum(Cost)
  FROM NACA NA
  WHERE NA.ID_TMS = :FRS_TMS_NO and NA.Main_Type = "AVDLR" and NA.BRANCH =
"AVIONICS" and NA.Date_Of_Trans >= :BeginDate
      and NA.Date_Of_Trans <= :EndDate
  INTO :FRSAVDLRAvionics;

  SELECT Sum(Cost)
  FROM NACA NA
  WHERE NA.ID_TMS = :FRS_TMS_NO and NA.Main_Type = "AVDLR" and NA.BRANCH =
"AIRFRAMES" and NA.Date_Of_Trans >= :BeginDate
      and NA.Date_Of_Trans <= :EndDate
  INTO :FRSAVDLRAirFrames;

  SELECT Sum(Cost)
  FROM NACA NA
  WHERE NA.ID_TMS = :FRS_TMS_NO and NA.Main_Type = "AVDLR" and NA.BRANCH =
"OTHER" and NA.Date_Of_Trans >= :BeginDate
      and NA.Date_Of_Trans <= :EndDate
  INTO :FRSAVDLROther;

  SELECT Sum(Cost)
  FROM  NACA NA
  WHERE NA.ID_TMS = :FRS_TMS_NO and NA.Main_Type = "AVDLR" and NA.BRANCH =
"MISC_TEC" and NA.Date_Of_Trans >= :BeginDate
      and NA.Date_Of_Trans <= :EndDate
  INTO :FRSAVDLRMiscTEC;
  SELECT Sum(Cost)
  FROM NACA NA
```

```
  WHERE NA.ID_TMS = :FRS_TMS_NO and NA.Main_Type = "AVDLR" and NA.BRANCH =
"OVERHEAD" and NA.Date_Of_Trans >= :BeginDate
      and NA.Date_Of_Trans <= :EndDate
  INTO :FRSAVDLROverhead;

  SELECT Sum(Cost)
  FROM NACA NA
  WHERE NA.ID_TMS = :FRS_TMS_NO and NA.Main_Type = "AFM" and NA.BRANCH =
"PWR_PLANTS" and NA.Date_Of_Trans >= :BeginDate
      and NA.Date_Of_Trans <= :EndDate
  INTO :FRSAFMPwrPlants;

  SELECT Sum(Cost)
  FROM NACA NA
  WHERE NA.ID_TMS = :FRS_TMS_NO and NA.Main_Type = "AFM" and NA.BRANCH =
"AVIONICS" and NA.Date_Of_Trans >= :BeginDate
      and NA.Date_Of_Trans <= :EndDate
  INTO :FRSAFMAvionics;

  SELECT Sum(Cost)
  FROM  NACA NA
  WHERE NA.ID_TMS = :FRS_TMS_NO and NA.Main_Type = "AFM" and NA.BRANCH =
"AIRFRAMES" and NA.Date_Of_Trans >= :BeginDate
      and NA.Date_Of_Trans <= :EndDate
  INTO :FRSAFMAirFrames;

  SELECT Sum(Cost)
  FROM NACA NA
  WHERE NA.ID_TMS = :FRS_TMS_NO and NA.Main_Type = "AFM" and NA.BRANCH = "OTHER"
and NA.Date_Of_Trans >= :BeginDate
      and NA.Date_Of_Trans <= :EndDate
  INTO :FRSAFMOther;

  SELECT Sum(Cost)
  FROM NACA NA
  WHERE NA.ID_TMS = :FRS_TMS_NO and NA.Main_Type = "AFM" and NA.BRANCH =
"MISC_TEC" and NA.Date_Of_Trans >= :BeginDate
      and NA.Date_Of_Trans <= :EndDate
  INTO :FRSAFMMiscTEC;

  SELECT Sum(Cost)
  FROM NACA NA
  WHERE NA.ID_TMS = :FRS_TMS_NO and NA.Main_Type = "AFM" and NA.BRANCH =
"OVERHEAD" and NA.Date_Of_Trans >= :BeginDate
      and NA.Date_Of_Trans <= :EndDate
  INTO :FRSAFMOverhead;

  SupportAVDLRPwrPlants = 0.0;
  SupportAVDLRAvionics = 0.0;
  SupportAVDLRAirFrames = 0.0;
  SupportAVDLROther = 0.0;
  SupportAVDLRMiscTEC = 0.0;
  SupportAVDLROverhead = 0.0;
  SupportAFMPwrPlants = 0.0;
```

```
  SupportAFMAvionics = 0.0;
  SupportAFMAirFrames = 0.0;
  SupportAFMOther = 0.0;
  SupportAFMMiscTEC = 0.0;
  SupportAFMOverhead = 0.0;

  FOR
    SELECT ID_TMS
    FROM TMS
    WHERE TMS = :TMSName and Mission = "S" and Fiscal_Year = :FY
    INTO :Support_TMS_NO
  DO
  BEGIN
    SELECT Sum(Cost)
    FROM NACA NA
    WHERE NA.ID_TMS = :Support_TMS_NO and NA.Main_Type = "AVDLR" and NA.BRANCH =
"PWR_PLANTS" and NA.Date_Of_Trans >= :BeginDate
      and NA.Date_Of_Trans <= :EndDate
    INTO :M;

    SELECT Sum(Cost)
    FROM NACA NA
    WHERE NA.ID_TMS = :Support_TMS_NO and NA.Main_Type = "AVDLR" and NA.BRANCH =
"AVIONICS" and NA.Date_Of_Trans >= :BeginDate
      and NA.Date_Of_Trans <= :EndDate
    INTO :N;

    SELECT Sum(Cost)
    FROM NACA NA
    WHERE NA.ID_TMS = :Support_TMS_NO and NA.Main_Type = "AVDLR" and NA.BRANCH =
"AIRFRAMES" and NA.Date_Of_Trans >= :BeginDate
      and NA.Date_Of_Trans <= :EndDate
    INTO :O;

    SELECT Sum(Cost)
    FROM NACA NA
    WHERE NA.ID_TMS = :Support_TMS_NO and NA.Main_Type = "AVDLR" and NA.BRANCH =
"OTHER" and NA.Date_Of_Trans >= :BeginDate
      and NA.Date_Of_Trans <= :EndDate
    INTO :P;

    SELECT Sum(Cost)
    FROM  NACA NA
    WHERE NA.ID_TMS = :Support_TMS_NO and NA.Main_Type = "AVDLR" and NA.BRANCH =
"MISC_TEC" and NA.Date_Of_Trans >= :BeginDate
      and NA.Date_Of_Trans <= :EndDate
    INTO :Q;

    SELECT Sum(Cost)
    FROM NACA NA
    WHERE NA.ID_TMS = :Support_TMS_NO and NA.Main_Type = "AVDLR" and NA.BRANCH =
"OVERHEAD" and NA.Date_Of_Trans >= :BeginDate
      and NA.Date_Of_Trans <= :EndDate
    INTO :R;
```

```
    SELECT Sum(Cost)
    FROM NACA NA
    WHERE NA.ID_TMS = :Support_TMS_NO and NA.Main_Type = "AFM" and NA.BRANCH =
"PWR_PLANTS" and NA.Date_Of_Trans >= :BeginDate
      and NA.Date_Of_Trans <= :EndDate
    INTO :S;

    SELECT Sum(Cost)
    FROM NACA NA
    WHERE NA.ID_TMS = :Support_TMS_NO and NA.Main_Type = "AFM" and NA.BRANCH =
"AVIONICS" and NA.Date_Of_Trans >= :BeginDate
      and NA.Date_Of_Trans <= :EndDate
    INTO :T;

    SELECT Sum(Cost)
    FROM  NACA NA
    WHERE NA.ID_TMS = :Support_TMS_NO and NA.Main_Type = "AFM" and NA.BRANCH =
"AIRFRAMES" and NA.Date_Of_Trans >= :BeginDate
      and NA.Date_Of_Trans <= :EndDate
    INTO :U;

    SELECT Sum(Cost)
    FROM NACA NA
    WHERE NA.ID_TMS = :Support_TMS_NO and NA.Main_Type = "AFM" and NA.BRANCH =
"OTHER" and NA.Date_Of_Trans >= :BeginDate
      and NA.Date_Of_Trans <= :EndDate
    INTO :V;

    SELECT Sum(Cost)
    FROM NACA NA
    WHERE NA.ID_TMS = :Support_TMS_NO and NA.Main_Type = "AFM" and NA.BRANCH =
"MISC_TEC" and NA.Date_Of_Trans >= :BeginDate
      and NA.Date_Of_Trans <= :EndDate
    INTO :W;

    SELECT Sum(Cost)
    FROM NACA NA
    WHERE NA.ID_TMS = :Support_TMS_NO and NA.Main_Type = "AFM" and NA.BRANCH =
"OVERHEAD" and NA.Date_Of_Trans >= :BeginDate
      and NA.Date_Of_Trans <= :EndDate
    INTO :X;

    SupportAVDLRPwrPlants =SupportAVDLRPwrPlants + M;
    SupportAVDLRAvionics = SupportAVDLRAvionics + N;
    SupportAVDLRAirFrames =SupportAVDLRAirFrames + O;
    SupportAVDLROther = SupportAVDLROther + P;
    SupportAVDLRMiscTEC = SupportAVDLRMiscTEC + Q;
    SupportAVDLROverhead = SupportAVDLROverhead + R;
    SupportAFMPwrPlants = SupportAFMPwrPlants + S;
    SupportAFMAvionics = SupportAFMAvionics + T;
    SupportAFMAirFrames = SupportAFMAirFrames + U;
    SupportAFMOther = SupportAFMOther + V;
    SupportAFMMiscTEC = SupportAFMMiscTEC + W;
```

```
    SupportAFMOverhead = SupportAFMOverhead + X;
  END

  TacairAVDLRPwrPlants = 0.0;
  TacairAVDLRAvionics = 0.0;
  TacairAVDLRAirFrames = 0.0;
  TacairAVDLROther = 0.0;
  TacairAVDLRMiscTEC = 0.0;
  TacairAVDLROverhead = 0.0;
  TacairAFMPwrPlants = 0.0;
  TacairAFMAvionics = 0.0;
  TacairAFMAirFrames = 0.0;
  TacairAFMOther = 0.0;
  TacairAFMMiscTEC = 0.0;
  TacairAFMOverhead = 0.0;

  FOR
    SELECT ID_TMS
    FROM TMS
    WHERE TMS = :TMSName and Mission = "T" and Fiscal_Year = :FY
    INTO :Tacair_TMS_NO
  DO
  BEGIN
    SELECT Sum(Cost)
    FROM NACA NA
    WHERE NA.ID_TMS = :Tacair_TMS_NO and NA.Main_Type = "AVDLR" and NA.BRANCH =
"PWR_PLANTS" and NA.Date_Of_Trans >= :BeginDate
      and NA.Date_Of_Trans <= :EndDate
    INTO :A;

    SELECT Sum(Cost)
    FROM NACA NA
    WHERE NA.ID_TMS = :Tacair_TMS_NO and NA.Main_Type = "AVDLR" and NA.BRANCH =
"AVIONICS" and NA.Date_Of_Trans >= :BeginDate
      and NA.Date_Of_Trans <= :EndDate
    INTO :B;

    SELECT Sum(Cost)
    FROM NACA NA
    WHERE NA.ID_TMS = :Tacair_TMS_NO and NA.Main_Type = "AVDLR" and NA.BRANCH =
"AIRFRAMES" and NA.Date_Of_Trans >= :BeginDate
      and NA.Date_Of_Trans <= :EndDate
    INTO :C;

    SELECT Sum(Cost)
    FROM NACA NA
    WHERE NA.ID_TMS = :Tacair_TMS_NO and NA.Main_Type = "AVDLR" and NA.BRANCH =
"OTHER" and NA.Date_Of_Trans >= :BeginDate
      and NA.Date_Of_Trans <= :EndDate
    INTO :D;

    SELECT Sum(Cost)
    FROM  NACA NA
```

```
   WHERE NA.ID_TMS = :Tacair_TMS_NO and NA.Main_Type = "AVDLR" and NA.BRANCH =
"MISC_TEC" and NA.Date_Of_Trans >= :BeginDate
      and NA.Date_Of_Trans <= :EndDate
   INTO :E;

   SELECT Sum(Cost)
   FROM NACA NA
   WHERE NA.ID_TMS = :Tacair_TMS_NO and NA.Main_Type = "AVDLR" and NA.BRANCH =
"OVERHEAD" and NA.Date_Of_Trans >= :BeginDate
      and NA.Date_Of_Trans <= :EndDate
   INTO :F;

   SELECT Sum(Cost)
   FROM NACA NA
   WHERE NA.ID_TMS = :Tacair_TMS_NO and NA.Main_Type = "AFM" and NA.BRANCH =
"PWR_PLANTS" and NA.Date_Of_Trans >= :BeginDate
      and NA.Date_Of_Trans <= :EndDate
   INTO :G;

   SELECT Sum(Cost)
   FROM NACA NA
   WHERE NA.ID_TMS = :Tacair_TMS_NO and NA.Main_Type = "AFM" and NA.BRANCH =
"AVIONICS" and NA.Date_Of_Trans >= :BeginDate
      and NA.Date_Of_Trans <= :EndDate
   INTO :H;

   SELECT Sum(Cost)
   FROM  NACA NA
   WHERE NA.ID_TMS = :Tacair_TMS_NO and NA.Main_Type = "AFM" and NA.BRANCH =
"AIRFRAMES" and NA.Date_Of_Trans >= :BeginDate
      and NA.Date_Of_Trans <= :EndDate
   INTO :I;

   SELECT Sum(Cost)
   FROM NACA NA
   WHERE NA.ID_TMS = :Tacair_TMS_NO and NA.Main_Type = "AFM" and NA.BRANCH =
"OTHER" and NA.Date_Of_Trans >= :BeginDate
      and NA.Date_Of_Trans <= :EndDate
   INTO :J;

   SELECT Sum(Cost)
   FROM NACA NA
   WHERE NA.ID_TMS = :Tacair_TMS_NO and NA.Main_Type = "AFM" and NA.BRANCH =
"MISC_TEC" and NA.Date_Of_Trans >= :BeginDate
      and NA.Date_Of_Trans <= :EndDate
   INTO :K;

   SELECT Sum(Cost)
   FROM NACA NA
   WHERE NA.ID_TMS = :Tacair_TMS_NO and NA.Main_Type = "AFM" and NA.BRANCH =
"OVERHEAD" and NA.Date_Of_Trans >= :BeginDate
      and NA.Date_Of_Trans <= :EndDate
   INTO :L;
   TacairAVDLRPwrPlants = TacairAVDLRPwrPlants + A;
```

```
    TacairAVDLRAvionics = TacairAVDLRAvionics + B;
    TacairAVDLRAirFrames = TacairAVDLRAirFrames + C;
    TacairAVDLROther = TacairAVDLROther + D;
    TacairAVDLRMiscTEC = TacairAVDLRMiscTEC + E;
    TacairAVDLROverhead = TacairAVDLROverhead + F;
    TacairAFMPwrPlants = TacairAFMPwrPlants + G;
    TacairAFMAvionics = TacairAFMAvionics + H;
    TacairAFMAirFrames = TacairAFMAirFrames + I;
    TacairAFMOther = TacairAFMOther + J;
    TacairAFMMiscTEC = TacairAFMMiscTEC + K;
    TacairAFMOverhead = TacairAFMOverhead + L;
  END

SUSPEND;
END ^
SET TERM ; ^
```

APPENDIX B. ASKIT REPORT, NACA EXTRACT, OP-20 REPORT

This appendix contains an example copy of an ASKIT report, NACA extract and a OP-20 report. These three reports are used by AIRPAC to allocate flight hour and maintenance funds, and track flight hour and maintenance expenditures.

A. ASKIT REPORT

The ASKIT report is utilized by AIRPAC to track reporting squadrons flight hour expenditures and fuel costs. The ASKIT report is received by AIRPAC as a typical naval message, with standard naval formatting of: From, To, Type of Classification, and Subject Line.

In the Remarks section the AIRPAC subordinate command provides the following information: AFAST / Command UIC / Type of Report / Begin Date / End Date.

There are three types of reports Update, Year-To-Date, and Initial Report. Initial reports are submitted by squadrons when their data is first entered into the database, Year-To-Date reports are submitted on an as needed basis (i.e., quality assurance check on data), and Update reports are submitted on a bi-monthly basis, itemizing flight hour and fuel costs during the last 15 day period.

The next two lines give command flight and fuel cost information in the following format: H or D (Home or Deployed), TMS, Flight Hours, Fuel Dollars, Month Flown.

For more information on the ASKIT message, see the COMNAVAIRPAC ASKIT Users Manual, September 1993.

Example of an ASKIT UPDATE Report

FM STRKFITRON ONE NINE TWO//
TO RUWFEAA/COMNAVAIRPAC SAN DIEGO CA//AFAST//
BT
UNCLAS //N07300//
MSGID/GENADMIN/STRKFITRON 192//
SUBJ/AFAST (3.1A) REPORT//
RMKS/ AFAST/R09076/UPDATE/96092/96106
H AMAF 0 0 APR
D AMAF 150 125041 APR
POC-VFA-192 LT M. R. CRAIG, MCO//
BT

B. NACA EXTRACT

NACA is an AIRPAC developed extract program which tracks the spending of Aviation Operations Maintenance (AOM) funds by extracting data from the NALCOMIS database. NACA provides a means of capturing the cost of requisitions as they are generated by the squadrons and AIMDs. This allows managers to monitor spending and plan maintenance based on funds available and historical cost of repairs. The information on each requisition is captured from NALCOMIS by an extract program and placed into two DOS text files, INPUT.DLY and INPUT.TO2 These files are then downloaded into the AFAST database where they are manipulated and finessed to provide the requisite data for the maintenance transaction report table. Detailed information concerning NACA and NALCOMIS can be obtained from the COMNAVAIRPAC CODE N422C4 NACA MANUAL [Boyd94].

1. INPUT.DLY DOS File

This DOS text file contains the extract data from the NALCOMIS requisition extract process and provides requisition data to AFAST II. A sample daily report item is

provided below. Note that data fields may be left blank as inputs are not always applicable in every maintenance transaction.

Example of a NACA INPUT.DLY Report and attribute definitions:

WC	
ORG	D9M
MCN	
JCN	P65269009
TRANS	
ACT	B
MAL	D13
PRI	03
WUC	51
SYSTEM	94265
COMPDATE	94277

WC	Work center code from completed MAF
ORG	Organization code of requisitioner
MCN	MAF control number
JCN	Job control number from MAF
TRANS	Transaction code from completed MAF
ACT	Action Take Code from MAF
MAL	Malfunction code from MAF
PRI	Priority of MAF and requisition
WUC	Work unit code from completed MAF
SYSTEM	System date from MAF
COMPDATE	Completed date from MAF

2. **INPUT.TO2 DOS File**

This DOS text file contains the extract data from the NALCOMIS requisition extract process and provides requisition data to AFAST II. Note that data fields may be left blank as inputs are not always applicable in every maintenance transaction.

Example of a NACA INPUT.TO2 Report and attribute definitions:

DDSN	4274G501
PROJ	AK0
PRI	03
STATUS	274COMPL
JCN	PC4273528
JON	YG501
QTY	1

NIIN	001651942
NOMEN	PACKING PREF
NETPRICE	.00
UNITPRICE	.23
EXTPRICE	.23
BADPRICE	
FGC	
COG	9Z
MCC	
REP_CON	C
ADV_CD	
TEC	AMAF
ORG	PC4
ORDERDATE	94274
PRICE	0.23
TDATE	94277

DDSN	Document date serial number from requisition
PROJ	Project code from requisition
PRI	Priority of MAF and requisition
STATUS	Supply status
JCN	Job Control Number from MAF
JON	Job order number from the requisition
QTY	Quantity ordered on requisition
NIIN	National item identification number
NOMEN	Nomenclature of item ordered
NETPRICE	Net price
UNITPRICE	Unit price from requisition
EXTPRICE	Extended price from the requisition
BADPRICE	Bad price indicator
FGC	Family group code; for repairables only
COG	Cognizance Code from requisition
MCC	Material control code from the requisition
REP_CON	Repairable/Consumable indicator
ADV_CD	Advice code from requisition
TEC	Type equipment code from requisition
ORG	Organization code of the requisitioner
ORDERDATE	Date requisition was ordered
PRICE	Numeric conversion of extended price data
TDATE	Transaction date, when data processed

C. OP-20 REPORT

OP-20 is an OPNAV document that is the basis for the Flight Hour Program budget. OP-20 depicts budgeted flight hours; a cost breakout for AFO, AVDLR, and all other AOM costs in terms of a projected average fleet wide Cost Per Flight Hour (CPFH);

and annual costs for each TMS assigned to specific program elements within budget activities. Two key factors dictate the level of AOM funds that AIRPAC receives: Budgeted Flight Hours and AOM CPH for each TMS aircraft. The major contributing factor that each NAS and CV controls in the flight hour program AOM budget is the AOM cost for each TMS aircraft. The major contributing factor that the squadron controls is the reported hours flown. . Detailed information concerning OP-20 accounting and reporting procedures can be obtained from the COMNAVAIRPAC INST 7305.1.

Example of an OP-20 Report and attribute definitions:

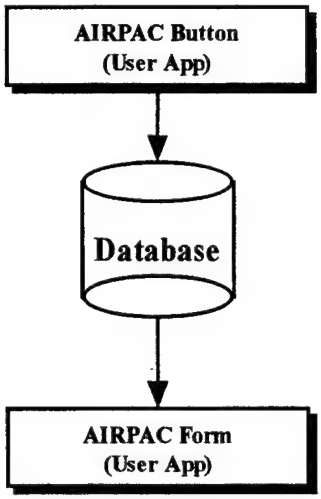
OP-20 REPORT

PROGRAM ELEMENT	TMS	FORCES	UTIL	HOURS	----COST PER HOUR---- FUEL	DLR	MNT	TOT
0204134M	A-6E	40.0	22.910	10997	820	1421	546	2787
TOTAL		40.0	22.910	10977	820	1421	546	2787
0204136N	FA-18C	130.0	29.581	46147	820	1210	640	2670
	FA-18A	10.0	29.450	3534	766	3886	1234	5886
TOTAL		140.0	29.572	69681	816	1400	682	2899
02041444N	F-14D	34.0	22.971	9372	998	1108	587	2693
	F-14A	56.0	23.073	15505	987	2376	1221	4584
TOTAL		90.0	23.034	24877	991	1898	982	3872

PROGRAM ELEMENT:	Identifies TMS specific mission type or Squadron. Examples: Marine TACAIR, Navy Support squadron VX-9, Navy FRS.
TMS	Type/Model/Series
FORCES	Number of aircraft maintained by CNAP
UTIL	Utilization . OPNAV constant used in determining number of flight hours to be allocated.
HOURS	Flight hours allocated for fiscal year
FUEL	OP-20 Fuel cost per flight hour
DLR	OP-20 AVDLR cost per flight hour
MNT	OP-20 AFM cost per flight hour

APPENDIX C. USER INTERFACE DIAGRAMS AND FORMS

A. AIRPAC FLOW DIAGRAM



Stored Procedures

- AIRPAC Summary, Should Cost

Input Parameters (AIRPAC Summary)

- Begin date, End Date, Fiscal Year

Input Parameters (Should Cost)

- Squadron UIC, Begin Date, End Date, Fiscal Year

Output Parameters (AIRPAC Summary)

- Budget AVDLR, Budget AFM, Budget Fuel, Budget Flt Hrs, Allocated AVDLR, Allocated AFM, Allocated Fuel, Allocated Flt Hrs, Executed AVDLR, Executed AFM, Executed Fuel, Executed Flt Hrs, Should AVDLR, Should AFM, Should Fuel

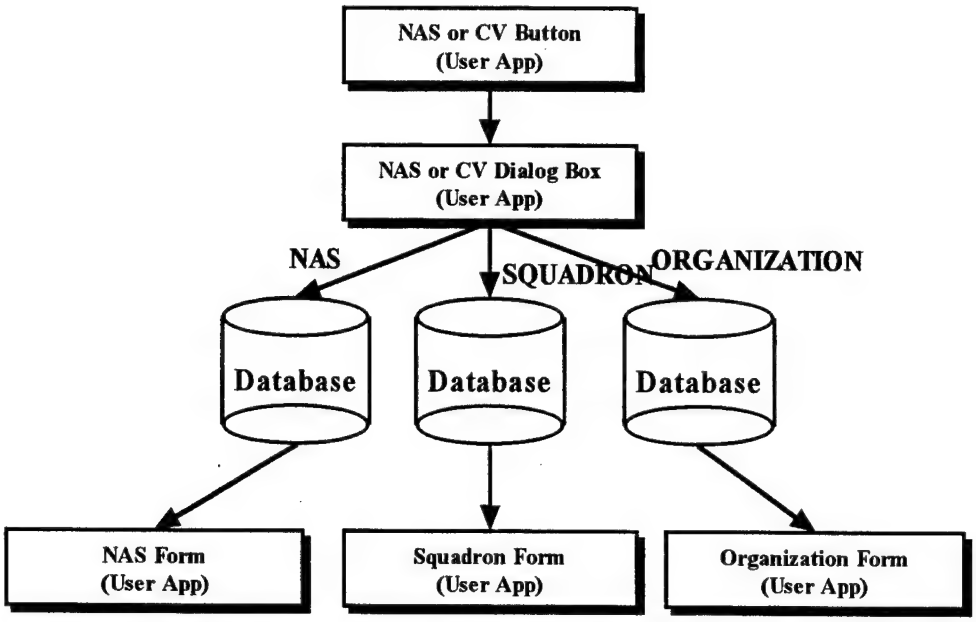
Output Parameters (Should Cost)

- Should AVDLR, Should AFM, Should Fuel

Calculated Parameters

- Budget Subtotal, Budget Total, Allocated Subtotal, Allocated Total, Executed Subtotal, Executed Total, Balance AVDLR, Balance AFM, Balance Subtotal, Balance Fuel, Balance Total, Balance Flt Hrs, Should Subtotal, Should Total, Delta AVDLR, Delta AFM, Delta Fuel, Delta Subtotal, Delta Total, Var AVDLR, Var AFM, Var Fuel, Var Subtotal, Var Total

B. NAS OR CV FLOW DIAGRAM



1. NAS or CV Form

Stored Procedures

- NAS_CV Summary, Should Cost

Input Parameters(NAS_CV Summary)

- NAS Name, Begin Date, End Date, Fiscal Year, Quarter

Input Parameters (Should Cost)

- Squadron UIC, Begin Date, End Date, Fiscal Year

Output Parameters (NAS_CV Summary)

- Allocated AVDLR, Allocated AFM, Allocated Fuel, Allocated Flt Hrs, Executed AVDLR, Executed AFM, Executed Fuel, Executed Flt Hrs, Should AVDLR, Should AFM, Should Fuel

Output Parameters (Should Cost)

- Should AVDLR, Should AFM, Should Fuel

Calculated Parameters:

- Allocated Subtotal, Allocated Total,Executed Subtotal, Executed Total, Balance AVDLR, Balance AFM, Balance Subtotal, Balance Fuel, Balance Total, Balance Flt Hrs,Should Subtotal, Should Total, Delta AVDLR, Delta AFM, Delta Fuel, Delta Subtotal, Delta Total Var AVDLR, Var AFM, Var Fuel, Var Subtotal, Var Total

2. Squadron Form from NAS or CV Dialog box

Stored Procedures

- NAS-Squadron Summary, NAS-Squadron Detail, Should Cost

Input Parameters (NAS-Squadron Summary)

- NAS or CV Name, Squadron Name, Begin Date, End Date, Fiscal Year, Quarter

Input Parameters (NAS-Squadron Detail)

- Squadron Name, NAS or CV Name, TMS Number, Fiscal Year, Begin Date, End Date

Input Parameters (Should Cost)

- Squadron UIC, Begin date, End Date, Fiscal Year

Output Parameters (NAS-Squadron Summary)

- Allocated Fuel, Allocated Flt Hrs, Executed AVDLR, Executed AFM, Executed Fuel, Executed Flt Hrs, Should AVDLR, Should AFM, Should Fuel

Output Parameters (NAS-Squadron Detail)

- Executed AVDLR, AVDLR Power Plants, AVDLR Avionics, AVDLR Air Frames, AVDLR Other, AVDLR Misc. TEC, AVDLR Overhead, Executed AFM, AFM Power Plants, AFM Avionics, AFM Air Frames, AFM Other, AFM Misc. TEC, AFM Overhead,Executed Fuel, Executed Flt Hrs, Should AVDLR, Should AFM, Should Fuel, Cost/Hour AVDLR, Cost/Hour AFM, Cost/Hour Fuel,Actual Cost/Hour AVDLR, Actual Cost/Hour AFM, Actual Cost/Hour Fuel

Output Parameters (Should Cost)

- Should AVDLR, Should AFM, Should Fuel

Calculated Parameters

- Allocated Total,Executed Subtotal, Executed Total, Balance AVDLR, Balance AFM, Balance Subtotal, Balance Fuel, Balance Total, Balance Flt Hrs,Should Subtotal, Should Total, Delta AVDLR, Delta AFM, Delta Fuel, Delta Subtotal, Delta Total Var AVDLR, Var AFM, Var Fuel, Var Subtotal, Var Total

3. Organization Form from NAS or CV Dialog Box

Stored Procedures

- ORG Summary

Input Parameters:

- ORG Name, NAS or CV Name, Begin date, End Date, Fiscal Year

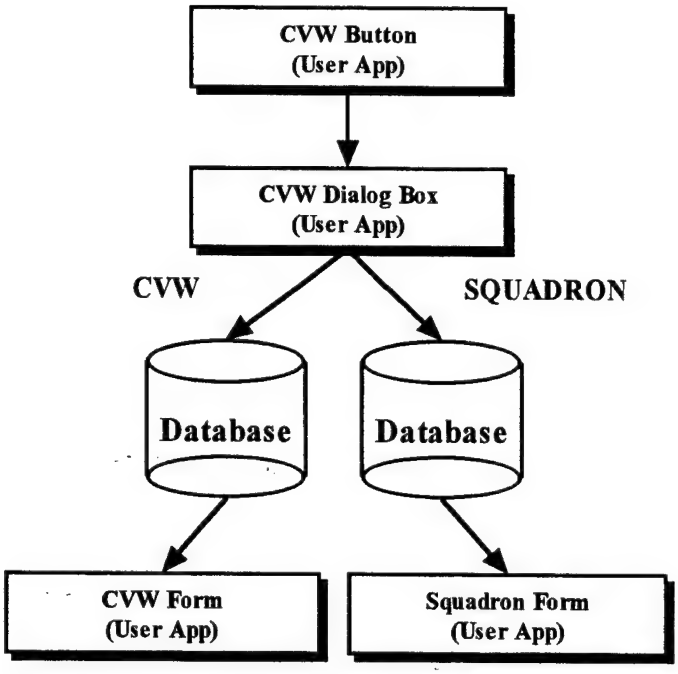
Output Parameters:

- AVDLR Power Plants, AVDLR Avionics, AVDLR Air Frames, AVDLR Other, AVDLR Misc. TEC, AVDLR Overhead, AFM Power Plants, AFM Avionics, AFM Air Frames, AFM Other, AFM Misc. TEC, AFM Overhead

Calculated Parameters:

- AVDLR Total, AFM Total

C. CVW FLOW DIAGRAM



1. CVW Form

Stored Procedures

- CVW Summary, Should Cost

Input Parameters (CVW Summary)

- CVW Name, Begin date, End Date, Fiscal Year, Quarter

Output Parameters (CVW Summary)

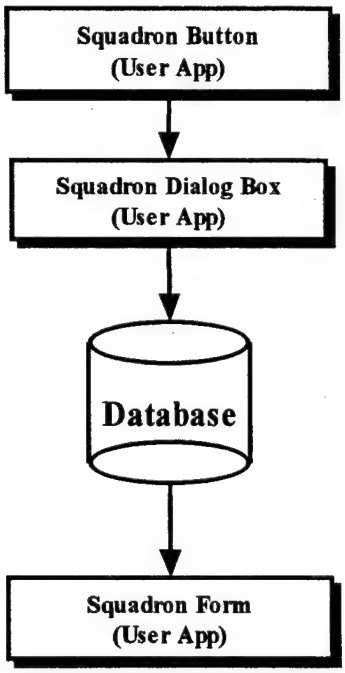
- Allocated Fuel, Allocated Flt Hrs, Executed AVDLR, Executed AFM, Executed Fuel, Executed Flt Hrs, Should AVDLR, Should AFM, Should Fuel

Calculated Parameters

- Allocated Total, Executed Subtotal, Executed Total, Balance AVDLR, Balance AFM, Balance Subtotal, Balance Fuel, Balance Total, Balance Flt Hrs, Should Subtotal, Should Total, Delta AVDLR, Delta AFM, Delta Fuel, Delta Subtotal, Delta Total Var AVDLR, Var AFM, Var Fuel, Var Subtotal, Var Total

2. Squadron Form from CVW Dialog Box (refer to section heading B-2 replacing references to NAS or CV with CVW)

D. SQUADRON FLOW DIAGRAM



Stored Procedures

- Squadron Summary, Squadron Detail, Should Cost

Input Parameters (Squadron Summary)

- Squadron Name, Begin date, End Date, Fiscal Year, Quarter

Input Parameters (Squadron Detail)

- Squadron Name, TMS Number, Begin date, End Date

Input Parameters (Should Cost)

- Squadron UIC, Begin date, End Date, Fiscal Year

Output Parameters (Squadron Summary)

- Allocated Fuel, Allocated Flt Hrs, Executed AVDLR, Executed AFM, Executed Fuel, Executed Flt Hrs, Should AVDLR, Should AFM, Should Fuel

Output Parameters (Squadron Detail)

- Executed AVDLR, AVDLR Power Plants, AVDLR Avionics, AVDLR Air Frames, AVDLR Other, AVDLR Misc. TEC, AVDLR Overhead, Executed AFM, AFM Power Plants, AFM Avionics, AFM Air Frames, AFM Other, AFM Misc. TEC, AFM Overhead,Executed Fuel, Executed Flt Hrs, Should AVDLR, Should AFM, Should Fuel, Cost/Hour AVDLR, Cost/Hour AFM, Cost/Hour Fuel,Actual Cost/Hour AVDLR, Actual Cost/Hour AFM, Actual Cost/Hour Fuel

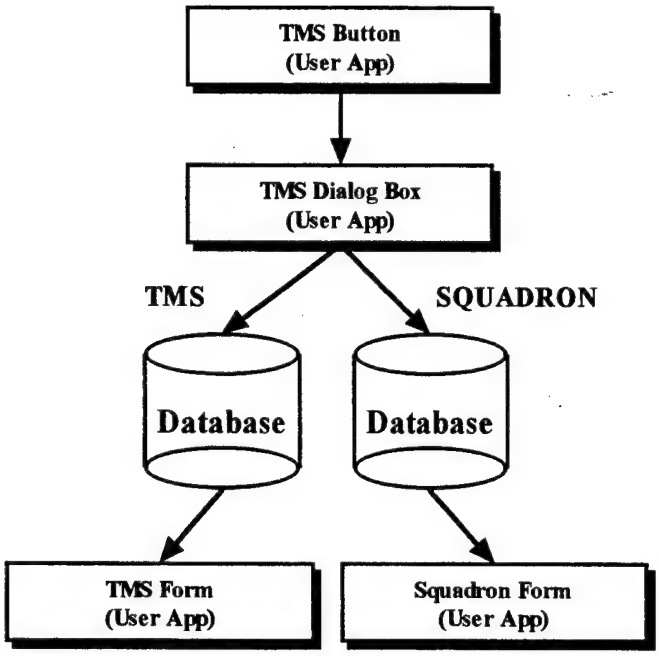
Output Parameters (Should Cost)

- Should AVDLR, Should AFM, Should Fuel

Calculated Parameters:

- Allocated Total, Executed Subtotal, Executed Total, Balance AVDLR, Balance AFM, Balance Subtotal, Balance Fuel, Balance Total, Balance Flt Hrs, Should Subtotal, Should Total, Delta AVDLR, Delta AFM, Delta Fuel, Delta Subtotal, Delta Total Var AVDLR, Var AFM, Var Fuel, Var Subtotal, Var Total

E. TMS FLOW DIAGRAM



1. TMS Form

Stored Procedures

- TMS Summary, TMS Detail,

Input Parameters (TMS Summary)

- TMS Name, Fiscal Year, Begin Date, End Date

Input Parameters (TMS Detail)

- TMS Name, Fiscal Year, Begin Date, End Date

Output Parameters (TMS Summary)

- FRS AVDLR, FRS AFM, FRS Fuel, FRS Flt Hrs, FRS Should AVDLR, FRS Should AFM, FRS Should Fuel, Support AVDLR, Support AFM, Support Fuel, Support Flt Hrs, Support Should AVDLR, Support Should AFM, Support Should Fuel, Tacair AVDLR, Tacair AFM, Tacair Fuel, Tacair Flt Hrs, Tacair Should AVDLR, Tacair Should AFM, Tacair Should Fuel

Output Parameters (TMS Detail)

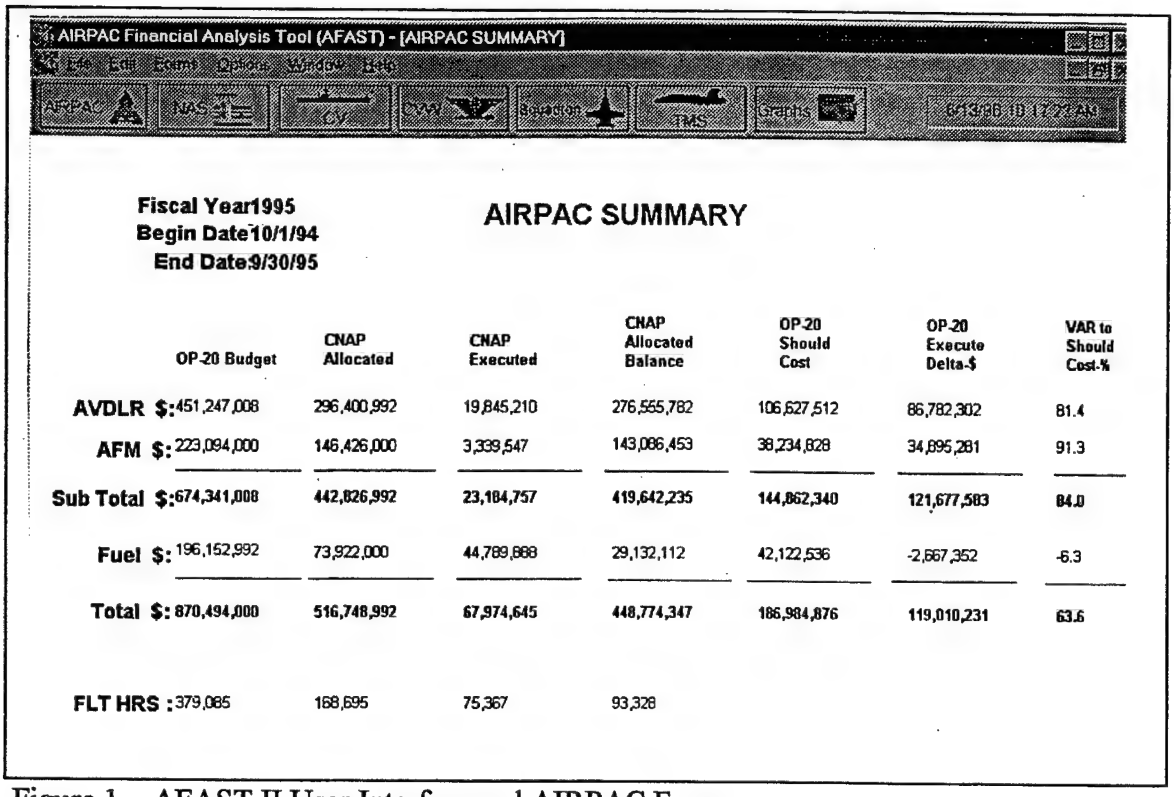
- FRS AVDLR Power Plants, FRS AVDLR Avionics, FRS AVDLR Air Frames, FRS AVDLR Other, FRS AVDLR Misc TEC, FRS AVDLR Overhead,FRS AFM Power Plants, FRS AFM Avionics, FRS AFM Air Frames, FRS AFM Other, FRS AFM Misc TEC, FRS AFM Overhead, Support AVDLR Power Plants, Support AVDLR Avionics, Support AVDLR Air Frames, Support AVDLR Other, Support AVDLR Misc TEC,

Support AVDLR Overhead, Support AFM Power Plants, Support AFM Avionics, Support AFM Air Frames, Support AFM Other, Support AFM Misc TEC, Support AFM Overhead, Tacair AVDLR Power Plants, Tacair AVDLR Avionics, Tacair AVDLR Air Frames, Tacair AVDLR Other, Tacair AVDLR Misc TEC, Tacair AVDLR Overhead, Tacair AFM Power Plants, Tacair AFM Avionics, Tacair AFM Air Frames, Tacair AFM Other, Tacair AFM Misc TEC, Tacair AFM Overhead

Calculated Parameters

- Total AVDLR, Total AFM, Total Fuel, Total Flt Hrs, Total Should AVDLR, Total Should AFM, Total Should Fuel, FRS Var AVDLR, FRS Var AFM, FRS Var Fuel, Support Var AVDLR, Support Var AFM, Support Var Fuel, Tacair Var AVDLR, Tacair Var AFM, Tacair Var Fuel, Total Var AVDLR, Total Var AFM, Total Var Fuel,Total AVDLR Power Plants, Total AVDLR Avionics, Total AVDLR Air Frames, Total AVDLR Other, Total AVDLR Misc TEC, Total AVDLR Overhead, Total AFM Power Plants, Total AFM Avionics, Total AFM Air Frames, Total AFM Other, Total AFM Misc TEC, Total AFM Overhead

2. Squadron Form from TMS Dialog Box (refer to section heading D)



	OP-20 Budget	CNAP Allocated	CNAP Executed	CNAP Allocated Balance	OP-20 Should Cost	OP-20 Execute Delta-$	VAR to Should Cost-%
AVDLR $:	451,247,008	296,400,992	19,845,210	276,555,782	106,627,512	86,782,302	81.4
AFM $:	223,094,000	146,426,000	3,339,547	143,086,453	38,234,828	34,895,281	91.3
Sub Total $:	**674,341,008**	**442,826,992**	**23,184,757**	**419,642,235**	**144,862,340**	**121,677,583**	**84.0**
Fuel $:	196,152,992	73,922,000	44,789,888	29,132,112	42,122,536	-2,667,352	-6.3
Total $:	**870,494,000**	**516,748,992**	**67,974,645**	**448,774,347**	**186,984,876**	**119,010,231**	**63.6**
FLT HRS :	379,085	168,695	75,367	93,328			

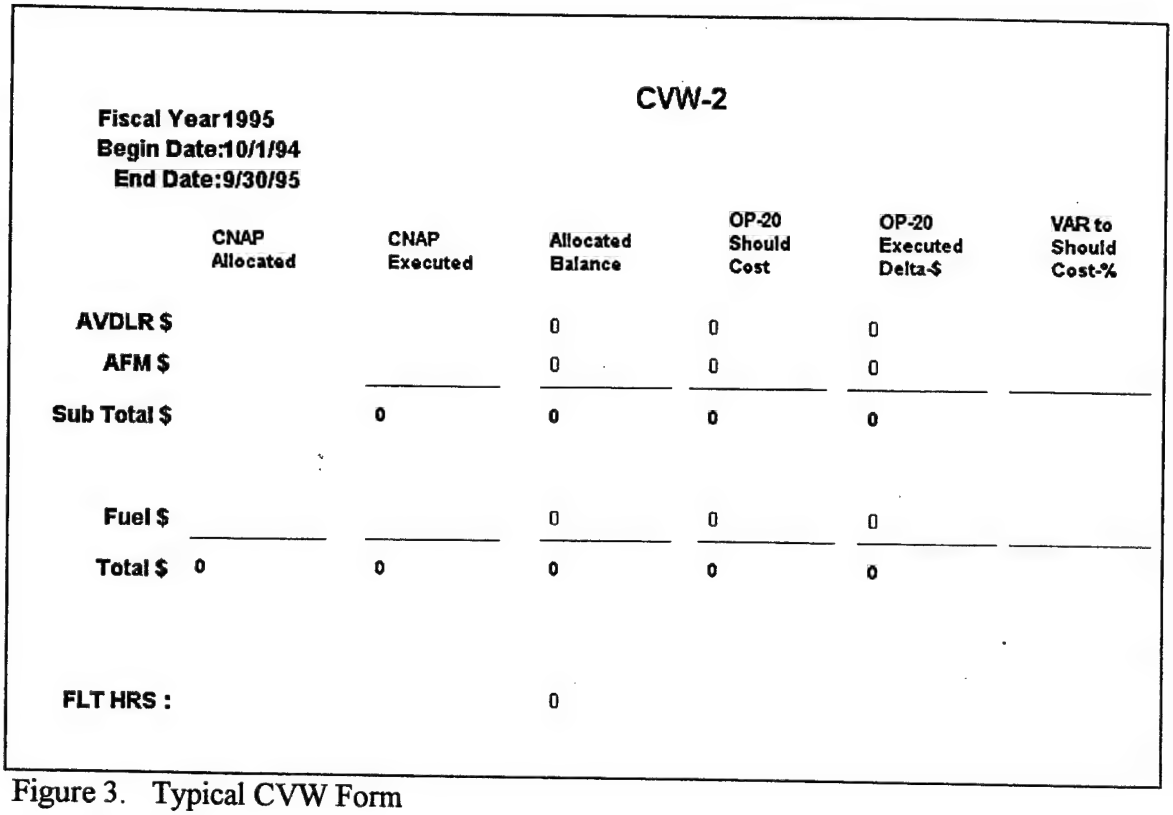
Figure 1. AFAST II User Interface and AIRPAC Form

Fiscal Year1995
Begin Date10/1/94
End Date:9/30/95

NAS North Island

	CNAP Allocated	CNAP Executed	Allocated Balance	OP-20 Should Cost	OP-20 Executed Delta-$	VAR to Should Cost-%
AVDLR $	76,865,000		76,865,000	40,836,168	40,836,168	100.0
AFM $	42,859,000		42,859,000	15,151,165	15,151,165	100.0
Sub Total $	**119,724,000**	**0**	**119,724,000**	**55,987,333**	**55,987,333**	**100.0**
Fuel $	9,510,178	8,888,562	621,616	8,081,273	-807,289	-10.0
Total $	**129,234,178**	**8,888,562**	**120,345,616**	**64,068,606**	**55,180,044**	**86.1**
FLT HRS :	34,467	26,922	7,545			

Figure 2. Typical NAS or CV Form

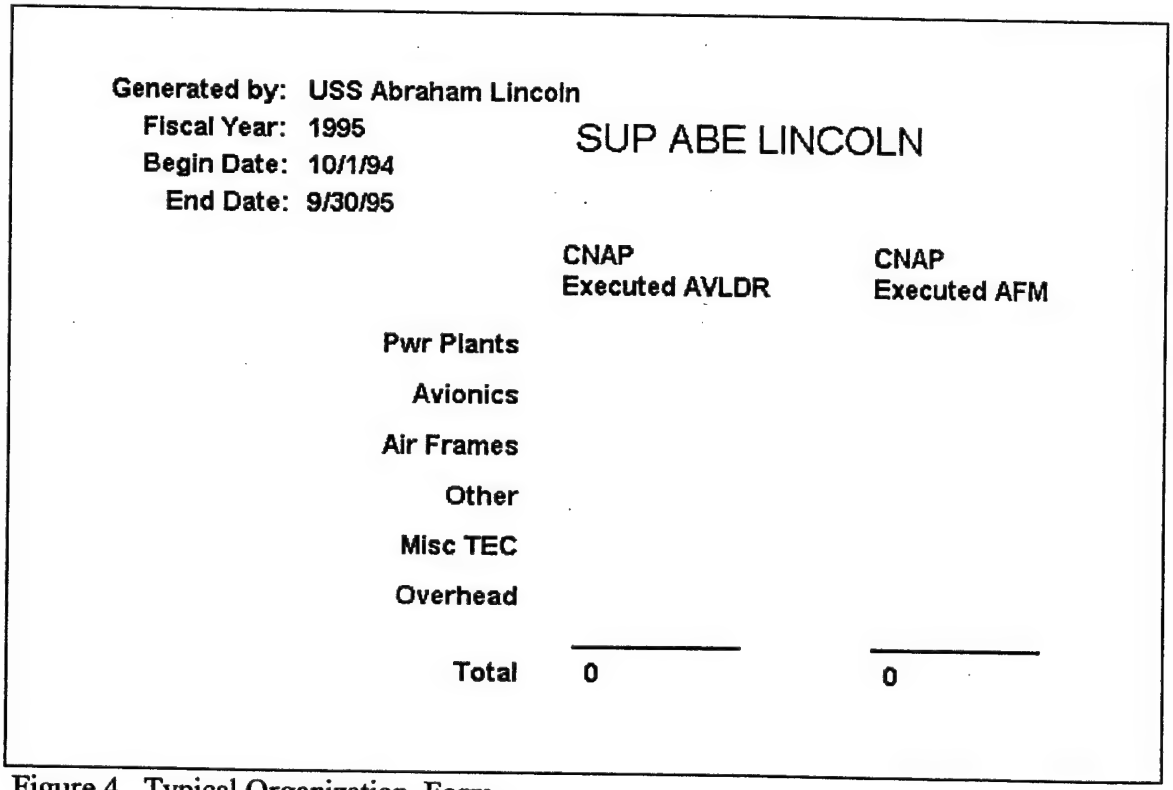


CVW-2

Fiscal Year1995
Begin Date:10/1/94
End Date:9/30/95

	CNAP Allocated	CNAP Executed	Allocated Balance	OP-20 Should Cost	OP-20 Executed Delta-$	VAR to Should Cost-%
AVDLR $			0	0	0	
AFM $			0	0	0	
Sub Total $		0	0	0	0	
Fuel $			0	0	0	
Total $	0	0	0	0	0	
FLT HRS :			0			

Figure 3. Typical CVW Form

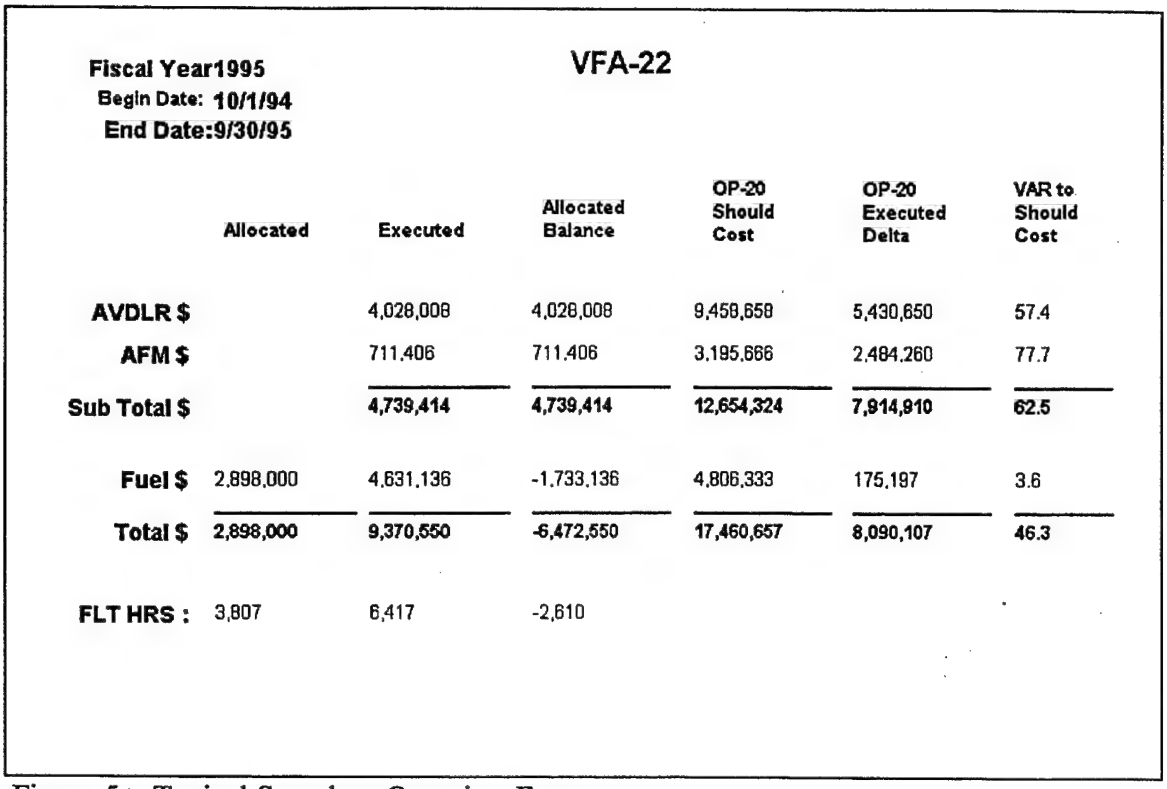


Generated by: USS Abraham Lincoln
Fiscal Year: 1995
Begin Date: 10/1/94
End Date: 9/30/95

SUP ABE LINCOLN

	CNAP Executed AVLDR	CNAP Executed AFM
Pwr Plants		
Avionics		
Air Frames		
Other		
Misc TEC		
Overhead		
Total	0	0

Figure 4. Typical Organization Form

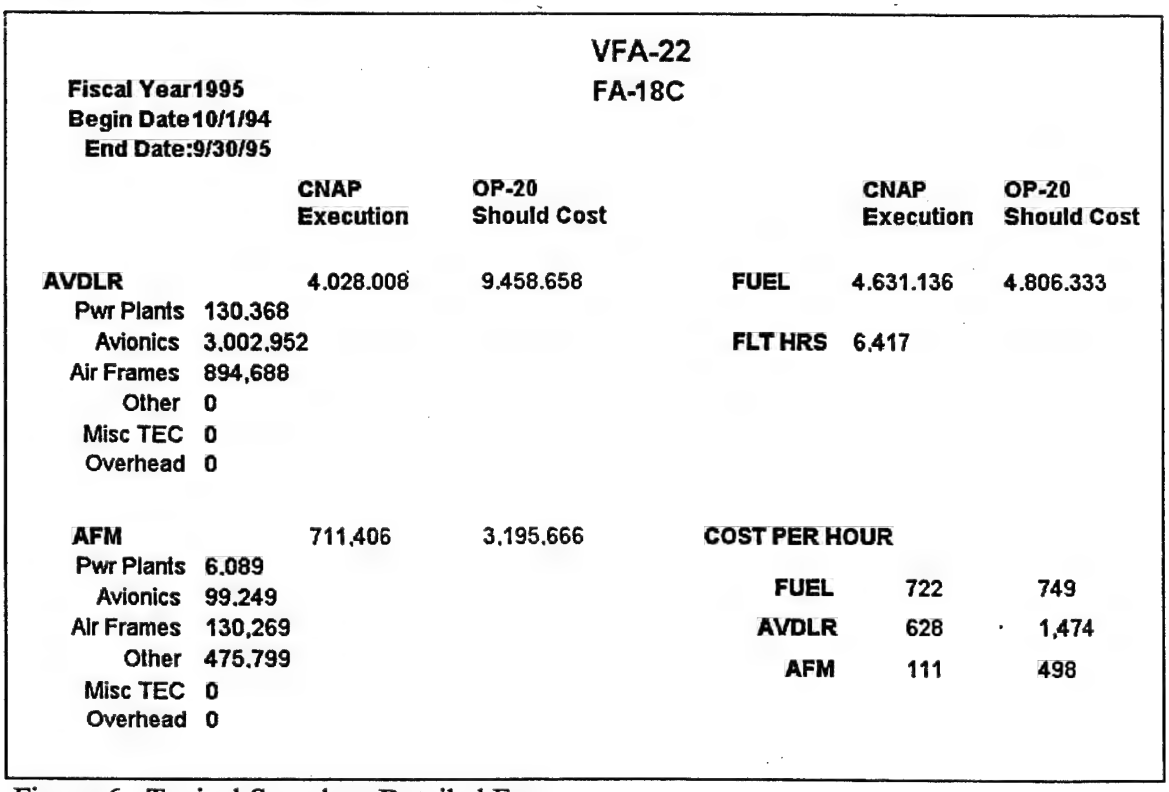


Fiscal Year1995
Begin Date: 10/1/94
End Date:9/30/95

VFA-22

	Allocated	Executed	Allocated Balance	OP-20 Should Cost	OP-20 Executed Delta	VAR to Should Cost
AVDLR $		4,028,008	4,028,008	9,458,658	5,430,650	57.4
AFM $		711,406	711,406	3,195,666	2,484,260	77.7
Sub Total $		4,739,414	4,739,414	12,654,324	7,914,910	62.5
Fuel $	2,898,000	4,631,136	-1,733,136	4,806,333	175,197	3.6
Total $	2,898,000	9,370,550	-6,472,550	17,460,657	8,090,107	46.3
FLT HRS :	3,807	6,417	-2,610			

Figure 5. Typical Squadron Overview Form

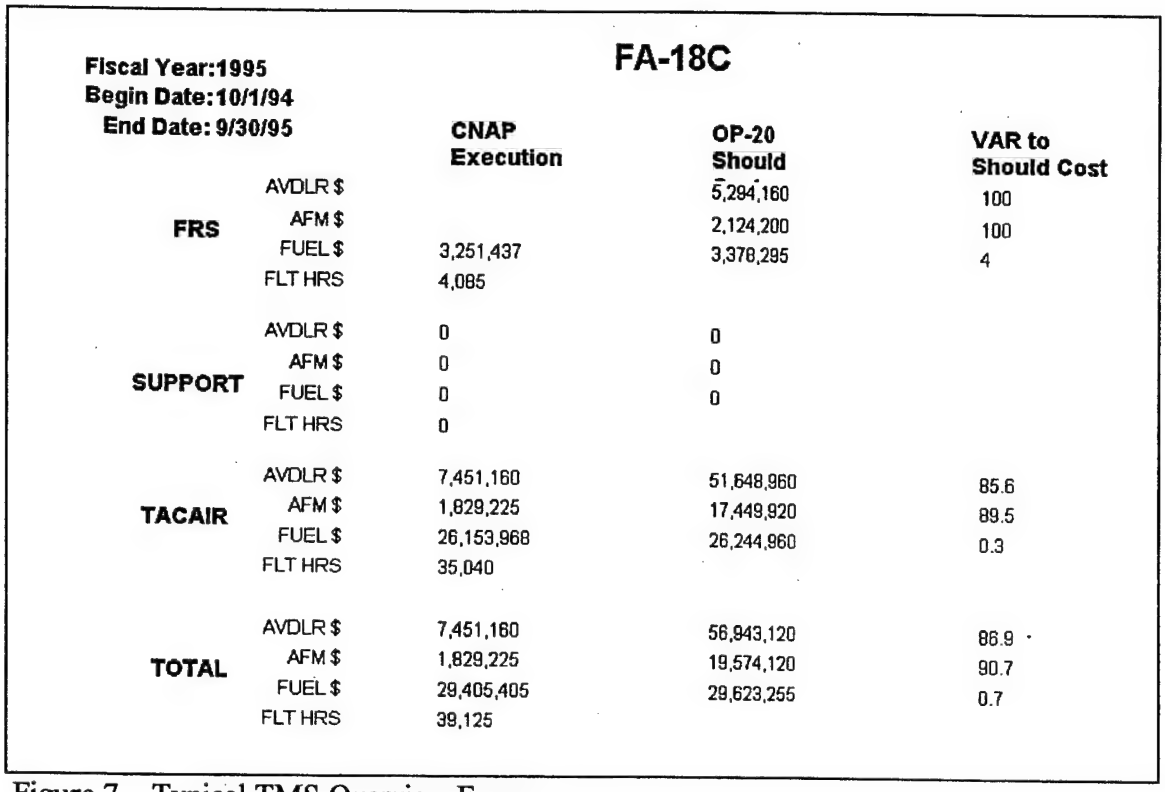


VFA-22
FA-18C

Fiscal Year1995
Begin Date10/1/94
End Date:9/30/95

		CNAP Execution	OP-20 Should Cost		CNAP Execution	OP-20 Should Cost
AVDLR		4.028.008	9.458.658	**FUEL**	4.631.136	4.806.333
Pwr Plants	130.368					
Avionics	3.002.952			**FLT HRS**	6.417	
Air Frames	894,688					
Other	0					
Misc TEC	0					
Overhead	0					
AFM		711,406	3,195,666	**COST PER HOUR**		
Pwr Plants	6.089					
Avionics	99.249			**FUEL**	722	749
Air Frames	130,269			**AVDLR**	628	1,474
Other	475.799			**AFM**	111	498
Misc TEC	0					
Overhead	0					

Figure 6. Typical Squadron Detailed Form

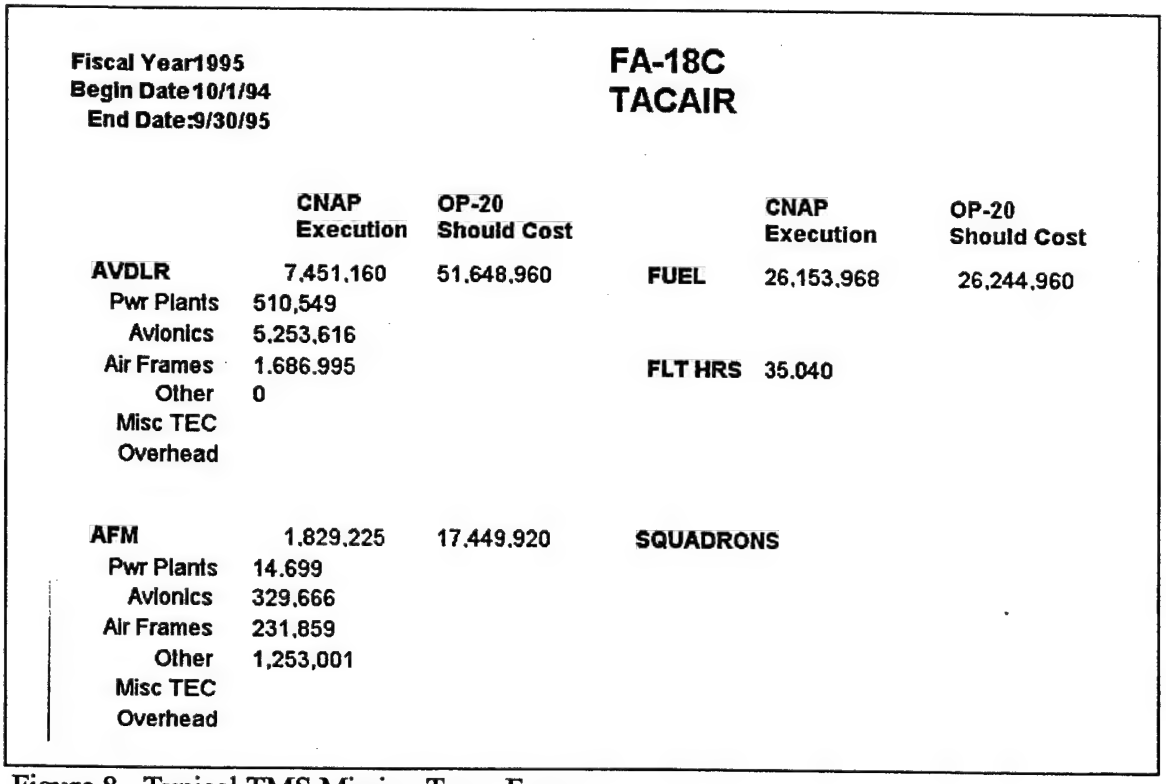


Fiscal Year:1995
Begin Date:10/1/94
End Date: 9/30/95

FA-18C

		CNAP Execution	OP-20 Should	VAR to Should Cost
FRS	AVDLR $		5,294,160	100
	AFM $		2,124,200	100
	FUEL $	3,251,437	3,378,295	4
	FLT HRS	4,085		
SUPPORT	AVDLR $	0	0	
	AFM $	0	0	
	FUEL $	0	0	
	FLT HRS	0		
TACAIR	AVDLR $	7,451,160	51,648,960	85.6
	AFM $	1,829,225	17,449,920	89.5
	FUEL $	26,153,968	26,244,960	0.3
	FLT HRS	35,040		
TOTAL	AVDLR $	7,451,160	56,943,120	86.9
	AFM $	1,829,225	19,574,120	90.7
	FUEL $	29,405,405	29,623,255	0.7
	FLT HRS	39,125		

Figure 7. Typical TMS Overview Form



Fiscal Year1995
Begin Date10/1/94
End Date:9/30/95

FA-18C TACAIR

	CNAP Execution	OP-20 Should Cost		CNAP Execution	OP-20 Should Cost
AVDLR	7,451,160	51,648,960	**FUEL**	26,153,968	26,244,960
Pwr Plants	510,549				
Avionics	5,253,616				
Air Frames	1,686,995		**FLT HRS**	35,040	
Other	0				
Misc TEC					
Overhead					
AFM	1,829,225	17,449,920	**SQUADRONS**		
Pwr Plants	14,699				
Avionics	329,666				
Air Frames	231,859				
Other	1,253,001				
Misc TEC					
Overhead					

Figure 8. Typical TMS Mission Type Form

APPENDIX D. ACRONYMS

AFAST	AIRPAC Financial Analysis Tool
AFM	Aircraft Fleet Maintenance
AFO	Aircraft Flight Operations
AIMD	Aircraft Intermediate Maintenance Department
AIRPAC	Naval Air Forces Pacific Fleet
ASKIT	Aviation Storekeeper Information Tracking System
ATAC	Advance Traceability and Control hub
AVDLR	Aviation Depot Level Repair
BCM	Beyond the Capability of Maintenance
BOR	Budget OPTAR Report
CINCPACFLT	Commander-in-Chief Pacific Fleet
CNAP	Commander, Naval Air Forces Pacific Fleet
CPH	Cost Per Hour
CV	Carrier
CVN	Carrier (nuclear)
CVW	Carrier Air Wing
DBMS	Database Management System
FHP	Flight Hour Program
FRS	Fleet Readiness Squadron
GUI	Graphical User Interface
IBM	International Business Machine

IMA	Intermediate Maintenance Activity
MAF	Maintenance Action Form
MDI	Multiple Document Interface
MIS	Management Information System
NACA	NALCOMIS AIMD Cost Accounting
NALCOMIS	Naval Aviation Logistics Command System
NAS	Naval Air Station
NON-RFI	Non-Ready For Issue
NPS	Naval Postgraduate School
OP-20	Operational Report 20
OPNAV	Office of the Chief of Naval Operations
OPTAR	Operating Targets
PC	Personal Computer
RFI	Ready For Issue
SDI	Single Document Interface
SFOEDL	Summary Filled Order Expenditure Difference Listing
TACAIR	Tactical Air (mission), Tactical Aircraft
TMS	Type/Model/Series
USMC	United States Marine Corps
USN	United States Navy
WSS	Wholesale Supply System
WYSIWYG	What You See Is What You Get

LIST OF REFERENCES

[Boyd94] Boyd, C.A., *COMNAVAIRPAC Fiscal Management Study,* Commander Naval Air Forces Pacific, 25 August 1994.

[CNAP86] COMNAVAIRPACINST 7305.1, *Instructions concerning Aircraft Operations Maintenance Funds,* Commander Naval Air Forces Pacific, 21 February 1986.

[CNAP93] *COMNAVAIRPAC ASKIT USERS MANUAL,* Commander Naval Air Forces Pacific, September 1993.

[Elmasri94] Elmasri, R., and Navathe, S., *Fundamentals of Database Systems,* Benjamin/Cummings Publishing Co., Inc. 1994.

[Fidel87] Fidel, Raya, *Database Design for Information Retrieval,* John Wiley and Sons, 1987.

[Gonzalez95] Gonzalez, Mark, personal correspondence with AIRPAC staff dtd 14 December 1995.

[Kroenke88] Kroenke, D., and Dolan, K., *Database Processing,* Macmillan Publishing Co., New York, NY, 1988.

[Lauff96] Lauff, Jeff, *AFAST II Design Review Minutes,* Naval Postgraduate School, Monterey, CA., 10 May 1996.

[Maciaszek89] Maciaszek, L., *Database Design and Implementation,* Prentice Hall, 1989.

[Marcus92] Marcus, A., *Graphic Design for Electronic Documents and User Interfaces,* ACM Press, New York, NY, 1992.

[Mayhew92] Mayhew, D., *Principles and Guidelines in Software User Interface Design,* Prentice Hall 1992.

[McCormick87] McCormick, B., Defanti, T., and Brown, R., "Visualization in scientific computing and computer graphics", *ACM SIGGRAPH 21,* November 1987.

[NPS96] Naval Postgraduate School, Research Proposal, March 21, 1996.

[Nelson80] Nelson, T., "Interactive systems and the design of virtuality", *Creative Computing,* November 1980 and December 1980.

[Nielsen90] Nielsen, J., *Hypertext and Hypermedia,* Academy Press, San Diego, CA, 1990.

[Norman91] Norman, K., *The Psychology of Menu Selection; Designing Cognitive Control at the Human/Computer Interface,* Ablex, Norwood, NJ, 1991.

[Pressman92] Pressman, R.S., *Software Engineering: A Practitioner's Approach*, McGraw Hill, New York, NY, 1992.

[Rutkowski82] Rutkowski, C., "An Introduction to the Human Applications Standard Computer Interface, Part 1; Theory and Principles", *Byte,* October 1982.

[Shneiderman93] Shneiderman, B., *Designing the User Interface, Strategies for Effective Human-Computer Interaction,* Addison-Wesley Publishing Co., 1993.

[Williams90] Williams, J., "Guidelines for Dialogue Design", *Designing and Using Human Computer Interface and Knowledge Based Systems,* Elsevier Science Publishers, 1990.

[Wu96] Wu, C. Thomas, *Advanced Database Systems (Class Notes - CS4312),* Department of Computer Science, Naval Postgraduate School, Monterey, CA., 1996.

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center 2
   8725 John J. Kingman Rd., STE 0944
   Ft. Belvoir, VA 22060-6218

2. Dudley Knox Library..... 2
   Naval Postgraduate School
   411 Dyer Rd.
   Monterey, California 93943-5101

3. Chairman, Code CS 1
   Computer Science Department
   Naval Postgraduate School
   Monterey, California 93943

4. Dr C. Thomas Wu, Code CS/KA 1
   Computer Science Department
   Naval Postgraduate School
   Monterey, California 93943

5. John Falby Code CS/FA 3
   Computer Science Department
   Naval Postgraduate School
   Monterey, California 93943

6. CDR. Mark J. Gonzalez 1
   Computer Science Department
   Naval Postgraduate School
   Monterey, California 93943

7. LT Mitch R. Hayes 1
   Computer Science Department
   Naval Postgraduate School
   Monterey, California 93943